US008625605B2

(12) **United States Patent**
Kastenholtz et al.

(10) **Patent No.: US 8,625,605 B2**
(45) **Date of Patent: Jan. 7, 2014**

(54) **NON-UNIFORM PER-PACKET PRIORITY MARKER FOR USE WITH ADAPTIVE PROTOCOLS**

(75) Inventors: **Frank Kastenholtz**, Medford, MA (US); **Laura Jane Poplawski Ma**, Somerville, MA (US); **Walter Clark Milliken**, Dover, NH (US); **Gregory Donald Troxel**, Stow, MA (US)

(73) Assignee: **Raytheon BBN Technologies Corp.**, Cambridge, MA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 242 days.

(21) Appl. No.: **13/112,257**

(22) Filed: **May 20, 2011**

(65) **Prior Publication Data**

US 2012/0176903 A1 Jul. 12, 2012

Related U.S. Application Data

(63) Continuation-in-part of application No. 12/200,264, filed on Aug. 28, 2008, now Pat. No. 8,203,956.

(51) **Int. Cl.**
H04L 12/56 (2011.01)

(52) **U.S. Cl.**
USPC **370/395.42**; 370/230.1; 370/231; 370/395.21

(58) **Field of Classification Search**
USPC 370/229, 230, 232, 237, 230.1, 231, 370/395.21, 395.42; 709/224
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

6,501,760	B1	12/2002	Ohba et al.	
6,646,988	B1 *	11/2003	Nandy et al.	370/235
6,781,956	B1	8/2004	Cheung	
6,885,638	B2 *	4/2005	Xu et al.	370/230
8,255,515	B1 *	8/2012	Melman et al.	709/224
2007/0127370	A1 *	6/2007	Chang et al.	370/229
2008/0175148	A1 *	7/2008	Todd et al.	370/235

OTHER PUBLICATIONS

International Search Report mailed Jul. 17, 2012 in corresponding International Application No. PCT/US2012/032050.

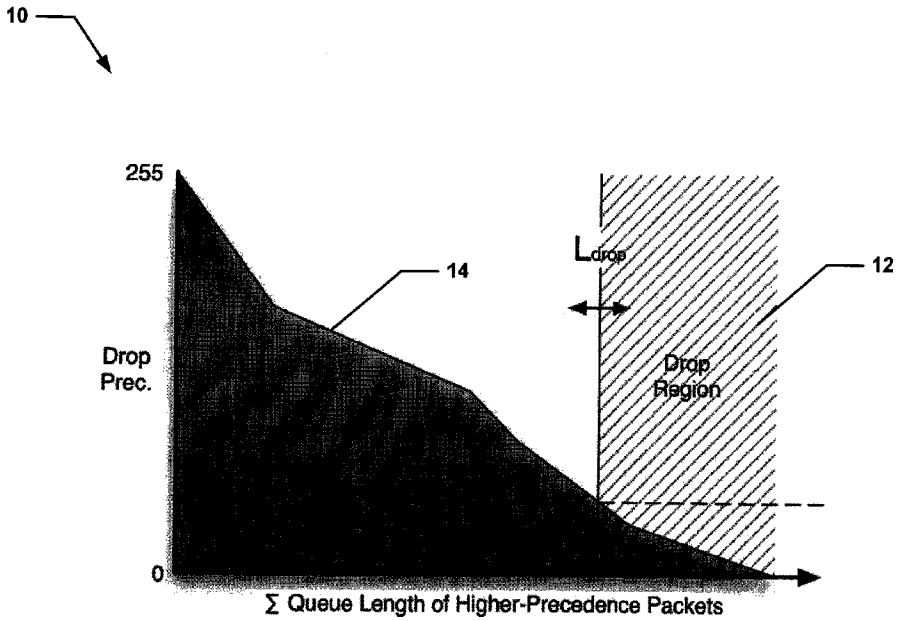
* cited by examiner

Primary Examiner — Brian D Nguyen
Assistant Examiner — Toan Nguyen
(74) *Attorney, Agent, or Firm* — Chapin IP Law, LLC

(57) **ABSTRACT**

A method, apparatus and computer program product for non-uniform per-packet priority marking for use with adaptive protocols is presented. A packet is received at a first network device, the packet assigned to a priority band. A priority is determined for the packet between a lowest priority of the priority band and a highest priority of the priority band, the priority for the packet selected based on a target distribution of priorities within the priority band, the target distribution comprising a distribution selected to achieve a desired capacity relationship among groups of packets assigned to different priority bands. The selected priority is assigned to the packet.

20 Claims, 23 Drawing Sheets



Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
07 JAN 2014		**00-00-2014 to 00-00-2014**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Non-Uniform Per-Packet Priority Marker for Use with Adaptive Protocols	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Raytheon BBN Technologies Corp,10 Moulton Street,Cambridge,MA,02138**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

A method, apparatus and computer program product for nonuniform per-packet priority marking for use with adaptive protocols is presented. A packet is received at a first network device, the packet assigned to a priority band. A priority is determined for the packet between a lowest priority of the priority band and a highest priority of the priority band, the priority for the packet selected based on a target distribution of priorities within the priority band, the target distribution comprising a distribution selected to achieve a desired capacity relationship among groups of packets assigned to different priority bands. The selected priority is assigned to the packet.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**35**	

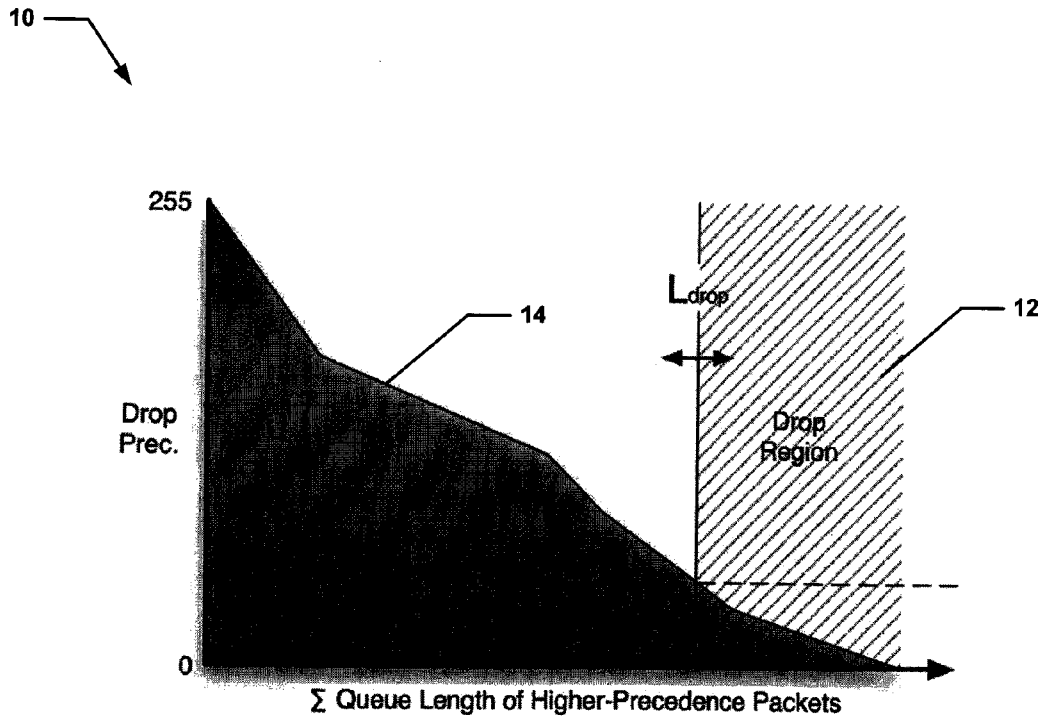

10
255
14
L_{drop}
12
Drop
Prec.
Drop
Region
0
Σ Queue Length of Higher-Precedence Packets

Figure 1

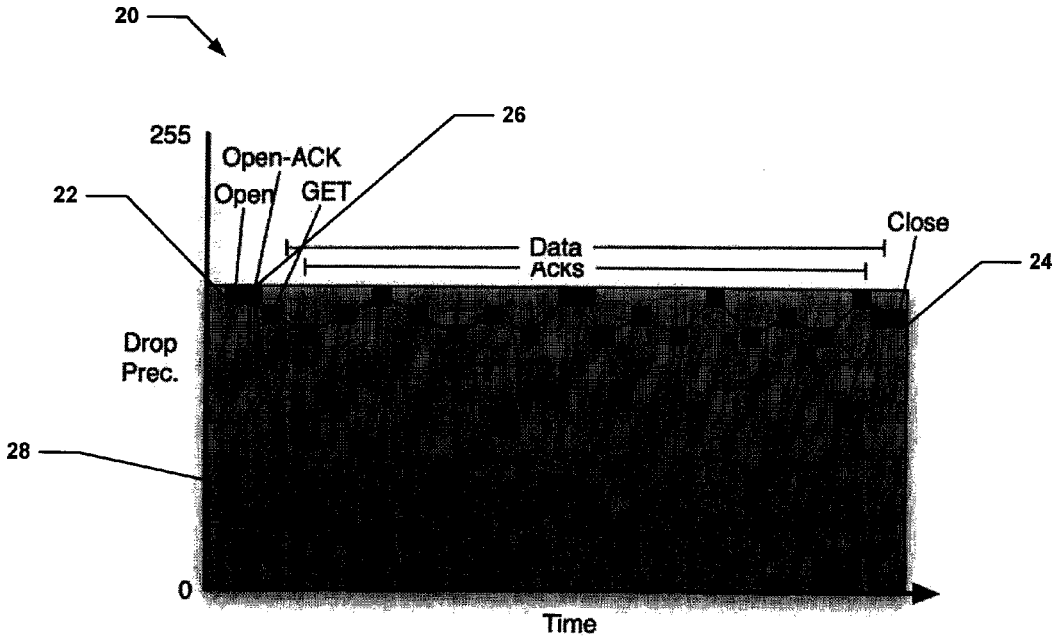

20
255
Open-ACK
Open
GET
26
22
Data
ACKs
Close
24
Drop
Prec.
28
0
Time

Figure 2

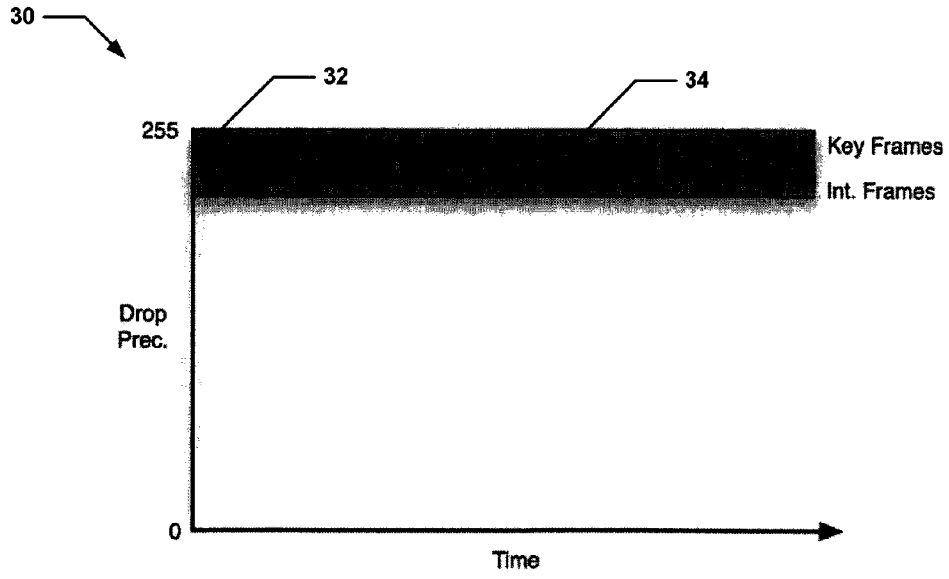


Figure 3

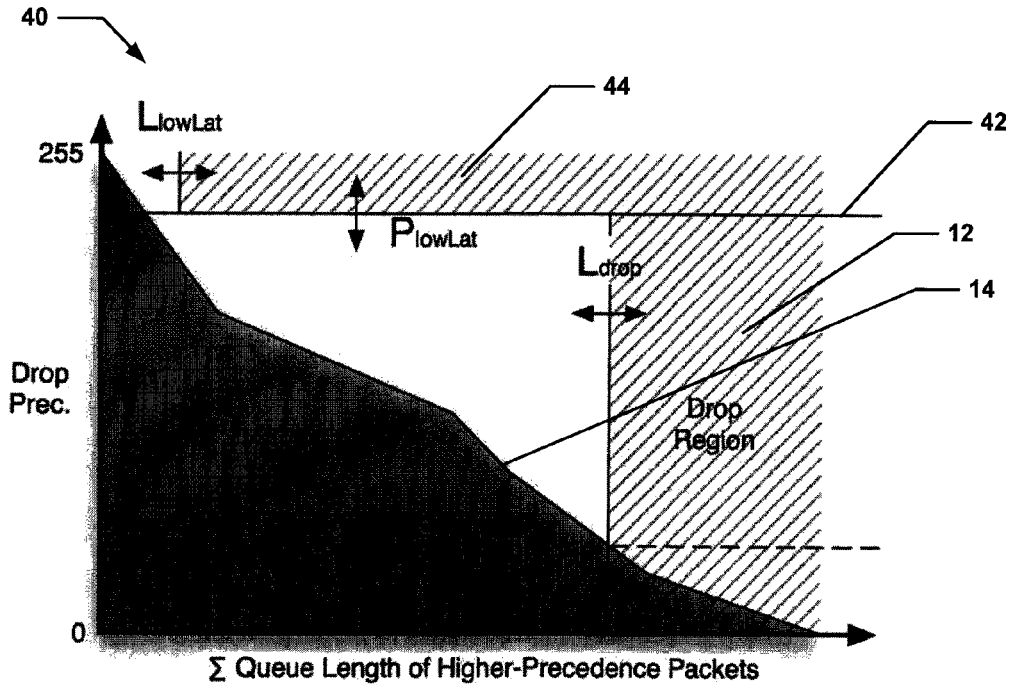
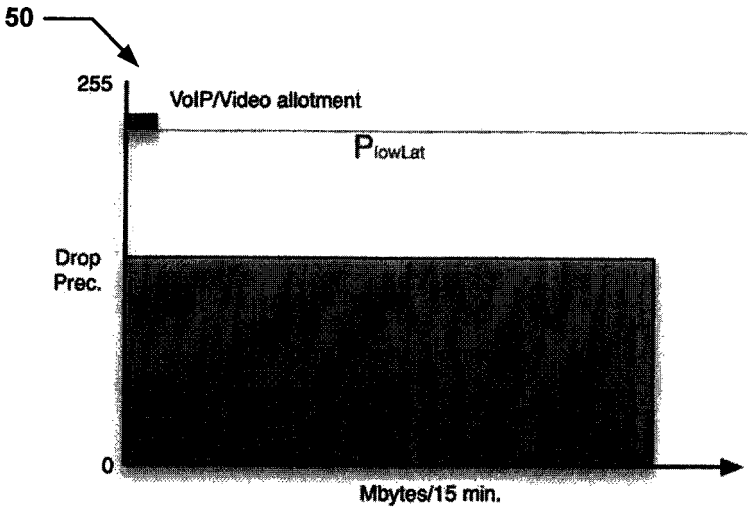
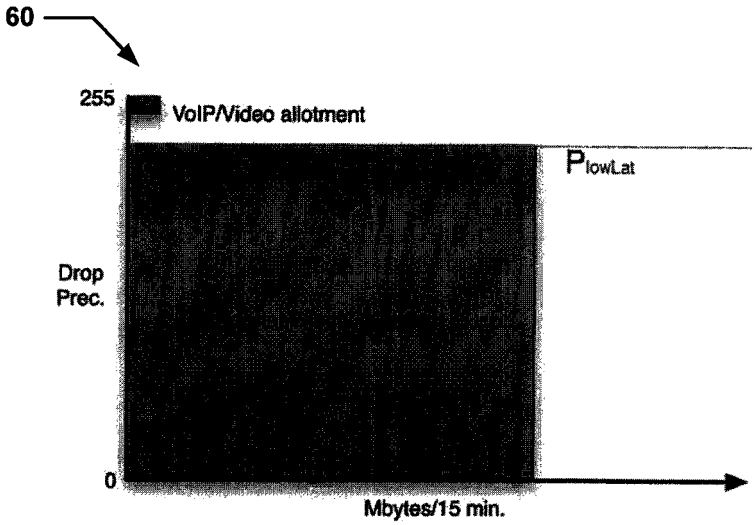

40
44
42
12
14
L_{lowLat}
255
P_{lowLat}
L_{drop}
Drop
Prec.
Drop
Region
0
Σ Queue Length of Higher-Precedence Packets

Figure 4



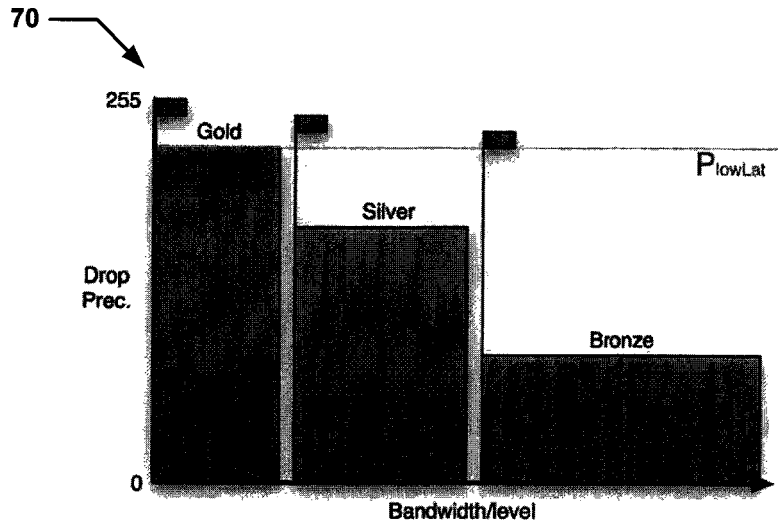
Example Simple SLA (w/low-latency)

Figure 5



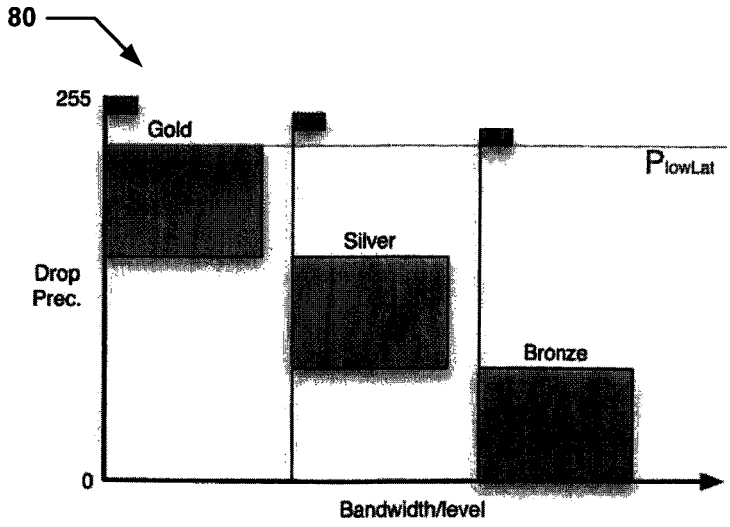
Example Preferred User SLA (w/low-latency)

Figure 6



Mapping 3-Tier User Class SLAs to PDQoS

Figure 7



3-Tier Pre-emption-based SLAs in PDQoS

Figure 8

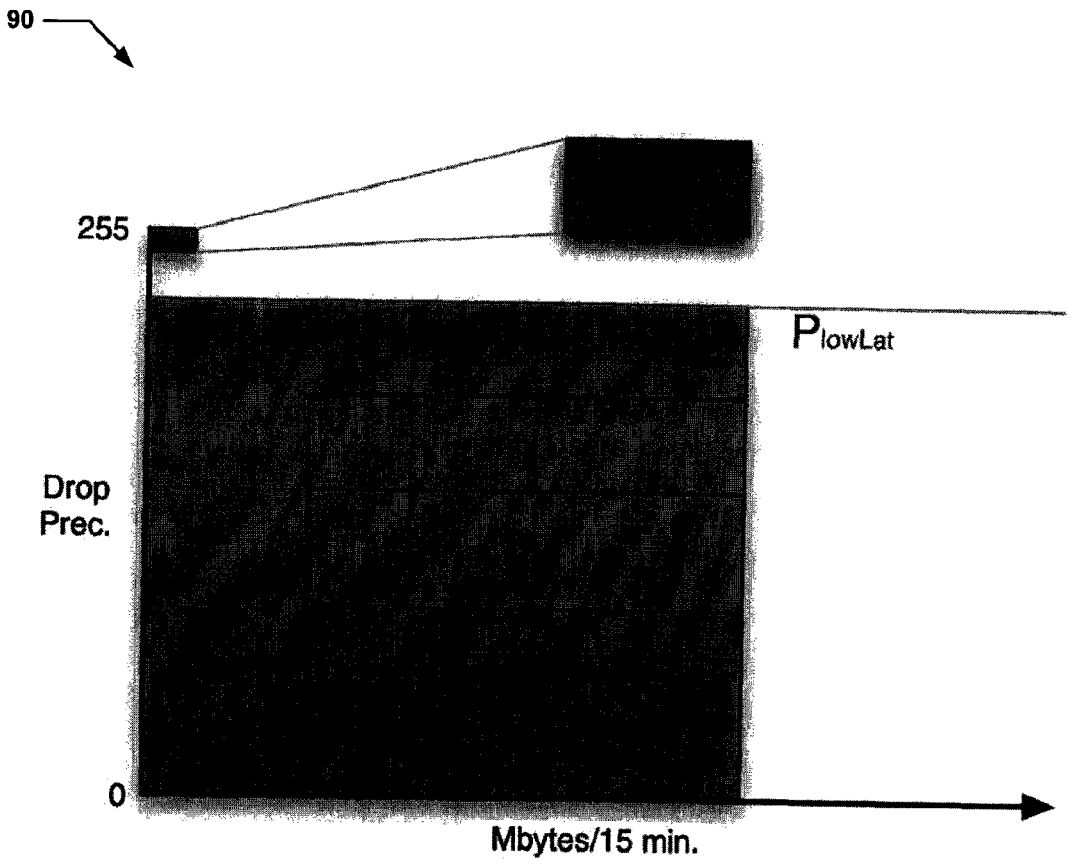


Figure 9

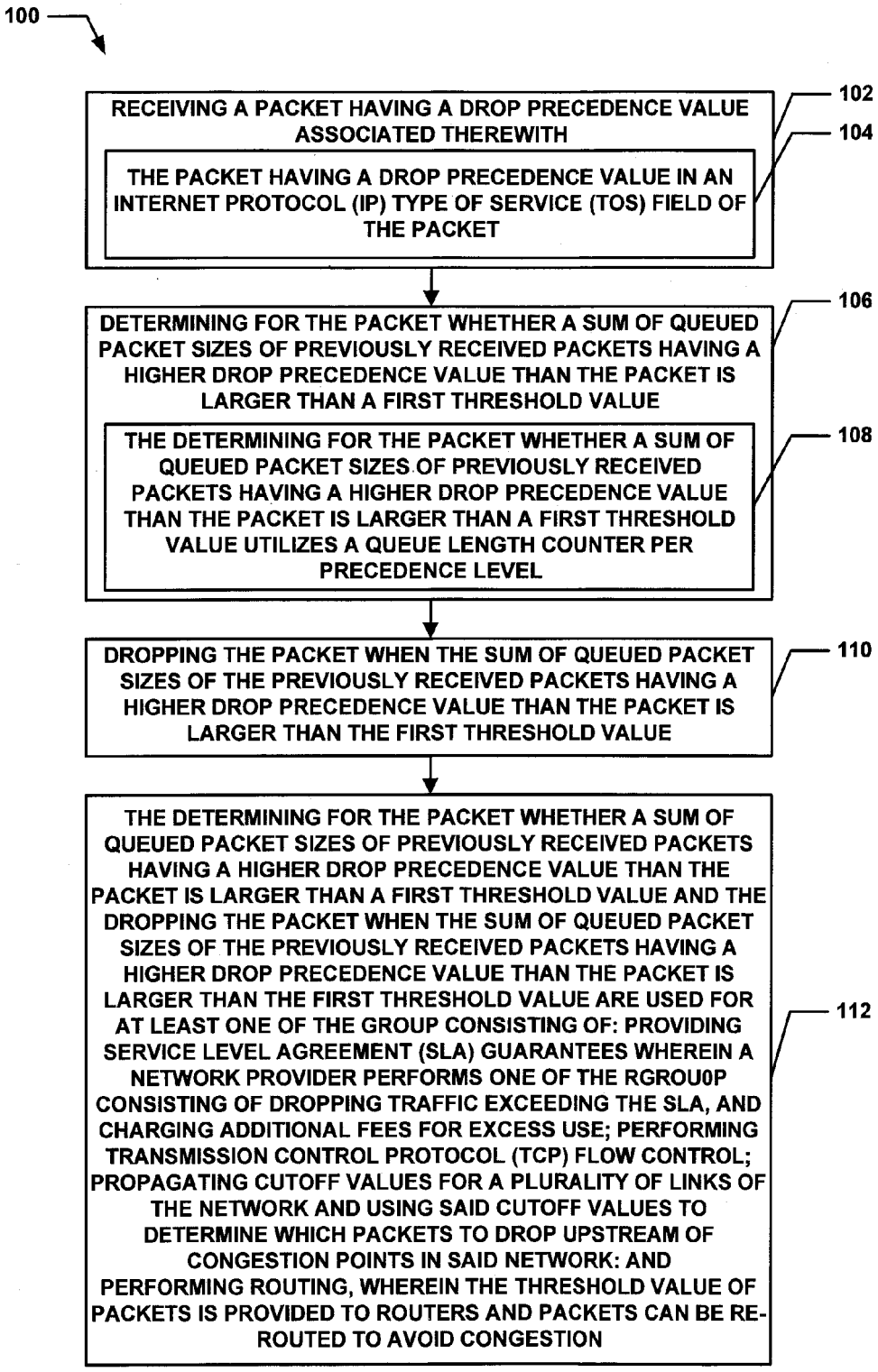

100
RECEIVING A PACKET HAVING A DROP PRECEDENCE VALUE ASSOCIATED THEREWITH
102
THE PACKET HAVING A DROP PRECEDENCE VALUE IN AN INTERNET PROTOCOL (IP) TYPE OF SERVICE (TOS) FIELD OF THE PACKET
104
DETERMINING FOR THE PACKET WHETHER A SUM OF QUEUED PACKET SIZES OF PREVIOUSLY RECEIVED PACKETS HAVING A HIGHER DROP PRECEDENCE VALUE THAN THE PACKET IS LARGER THAN A FIRST THRESHOLD VALUE
106
THE DETERMINING FOR THE PACKET WHETHER A SUM OF QUEUED PACKET SIZES OF PREVIOUSLY RECEIVED PACKETS HAVING A HIGHER DROP PRECEDENCE VALUE THAN THE PACKET IS LARGER THAN A FIRST THRESHOLD VALUE UTILIZES A QUEUE LENGTH COUNTER PER PRECEDENCE LEVEL
108
DROPPING THE PACKET WHEN THE SUM OF QUEUED PACKET SIZES OF THE PREVIOUSLY RECEIVED PACKETS HAVING A HIGHER DROP PRECEDENCE VALUE THAN THE PACKET IS LARGER THAN THE FIRST THRESHOLD VALUE
110
THE DETERMINING FOR THE PACKET WHETHER A SUM OF QUEUED PACKET SIZES OF PREVIOUSLY RECEIVED PACKETS HAVING A HIGHER DROP PRECEDENCE VALUE THAN THE PACKET IS LARGER THAN A FIRST THRESHOLD VALUE AND THE DROPPING THE PACKET WHEN THE SUM OF QUEUED PACKET SIZES OF THE PREVIOUSLY RECEIVED PACKETS HAVING A HIGHER DROP PRECEDENCE VALUE THAN THE PACKET IS LARGER THAN THE FIRST THRESHOLD VALUE ARE USED FOR AT LEAST ONE OF THE GROUP CONSISTING OF: PROVIDING SERVICE LEVEL AGREEMENT (SLA) GUARANTEES WHEREIN A NETWORK PROVIDER PERFORMS ONE OF THE RGROU0P CONSISTING OF DROPPING TRAFFIC EXCEEDING THE SLA, AND CHARGING ADDITIONAL FEES FOR EXCESS USE; PERFORMING TRANSMISSION CONTROL PROTOCOL (TCP) FLOW CONTROL; PROPAGATING CUTOFF VALUES FOR A PLURALITY OF LINKS OF THE NETWORK AND USING SAID CUTOFF VALUES TO DETERMINE WHICH PACKETS TO DROP UPSTREAM OF CONGESTION POINTS IN SAID NETWORK: AND PERFORMING ROUTING, WHEREIN THE THRESHOLD VALUE OF PACKETS IS PROVIDED TO ROUTERS AND PACKETS CAN BE RE-ROUTED TO AVOID CONGESTION
112

Figure 10A

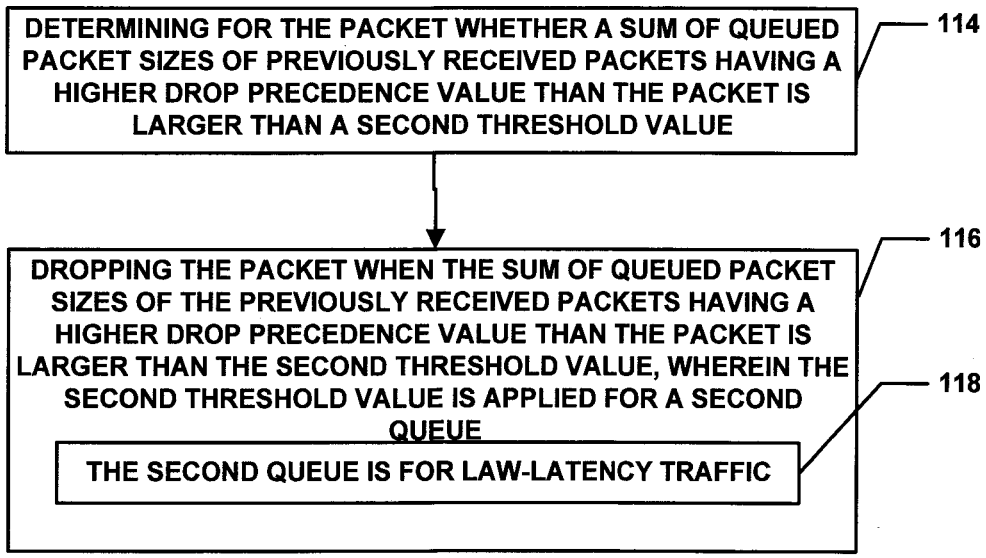

DETERMINING FOR THE PACKET WHETHER A SUM OF QUEUED PACKET SIZES OF PREVIOUSLY RECEIVED PACKETS HAVING A HIGHER DROP PRECEDENCE VALUE THAN THE PACKET IS LARGER THAN A SECOND THRESHOLD VALUE
114
DROPPING THE PACKET WHEN THE SUM OF QUEUED PACKET SIZES OF THE PREVIOUSLY RECEIVED PACKETS HAVING A HIGHER DROP PRECEDENCE VALUE THAN THE PACKET IS LARGER THAN THE SECOND THRESHOLD VALUE, WHEREIN THE SECOND THRESHOLD VALUE IS APPLIED FOR A SECOND QUEUE
116
THE SECOND QUEUE IS FOR LAW-LATENCY TRAFFIC
118

Figure 10B

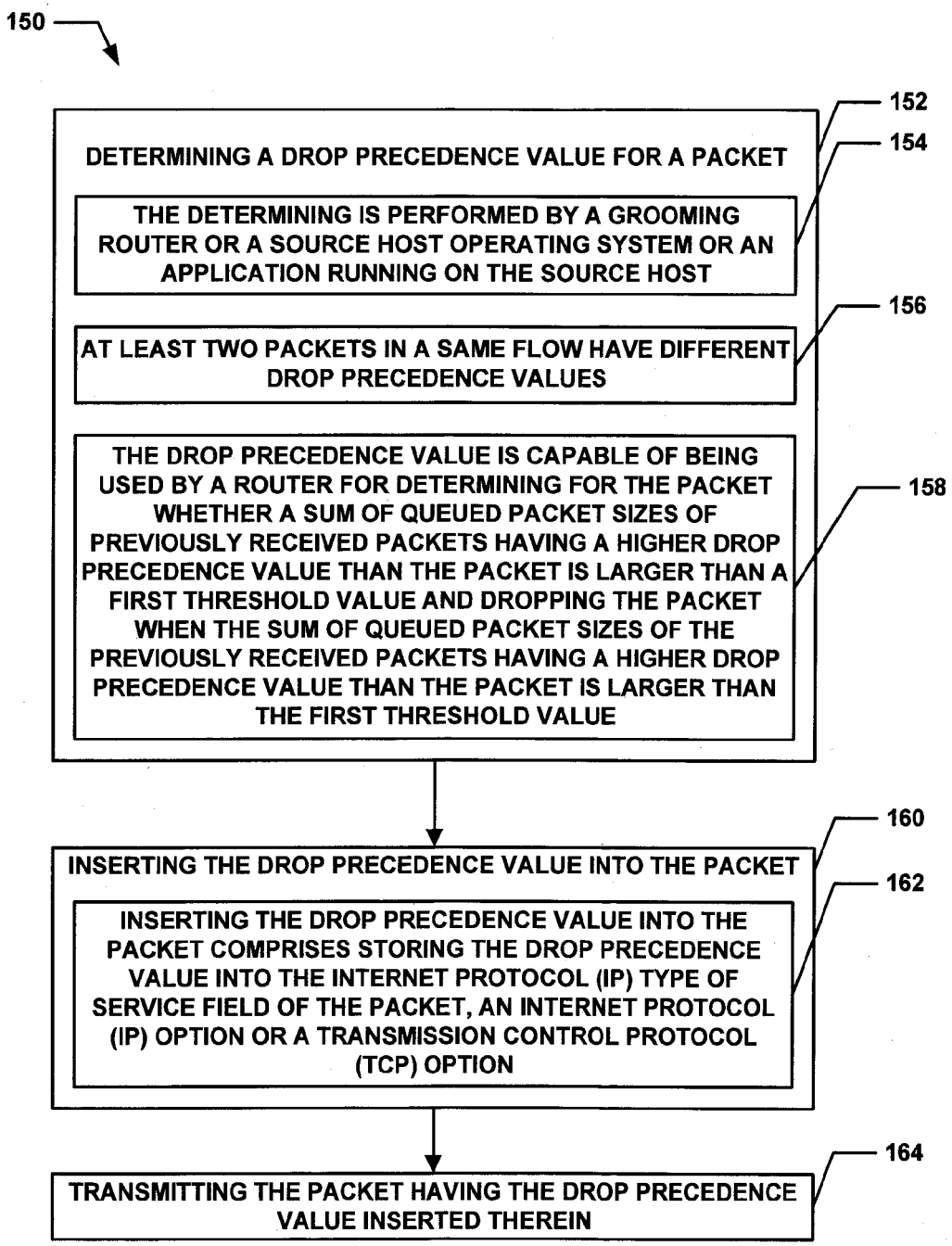

150
152
154
DETERMINING A DROP PRECEDENCE VALUE FOR A PACKET
THE DETERMINING IS PERFORMED BY A GROOMING ROUTER OR A SOURCE HOST OPERATING SYSTEM OR AN APPLICATION RUNNING ON THE SOURCE HOST
156
AT LEAST TWO PACKETS IN A SAME FLOW HAVE DIFFERENT DROP PRECEDENCE VALUES
158
THE DROP PRECEDENCE VALUE IS CAPABLE OF BEING USED BY A ROUTER FOR DETERMINING FOR THE PACKET WHETHER A SUM OF QUEUED PACKET SIZES OF PREVIOUSLY RECEIVED PACKETS HAVING A HIGHER DROP PRECEDENCE VALUE THAN THE PACKET IS LARGER THAN A FIRST THRESHOLD VALUE AND DROPPING THE PACKET WHEN THE SUM OF QUEUED PACKET SIZES OF THE PREVIOUSLY RECEIVED PACKETS HAVING A HIGHER DROP PRECEDENCE VALUE THAN THE PACKET IS LARGER THAN THE FIRST THRESHOLD VALUE
160
INSERTING THE DROP PRECEDENCE VALUE INTO THE PACKET
162
INSERTING THE DROP PRECEDENCE VALUE INTO THE PACKET COMPRISES STORING THE DROP PRECEDENCE VALUE INTO THE INTERNET PROTOCOL (IP) TYPE OF SERVICE FIELD OF THE PACKET, AN INTERNET PROTOCOL (IP) OPTION OR A TRANSMISSION CONTROL PROTOCOL (TCP) OPTION
164
TRANSMITTING THE PACKET HAVING THE DROP PRECEDENCE VALUE INSERTED THEREIN

Figure 11

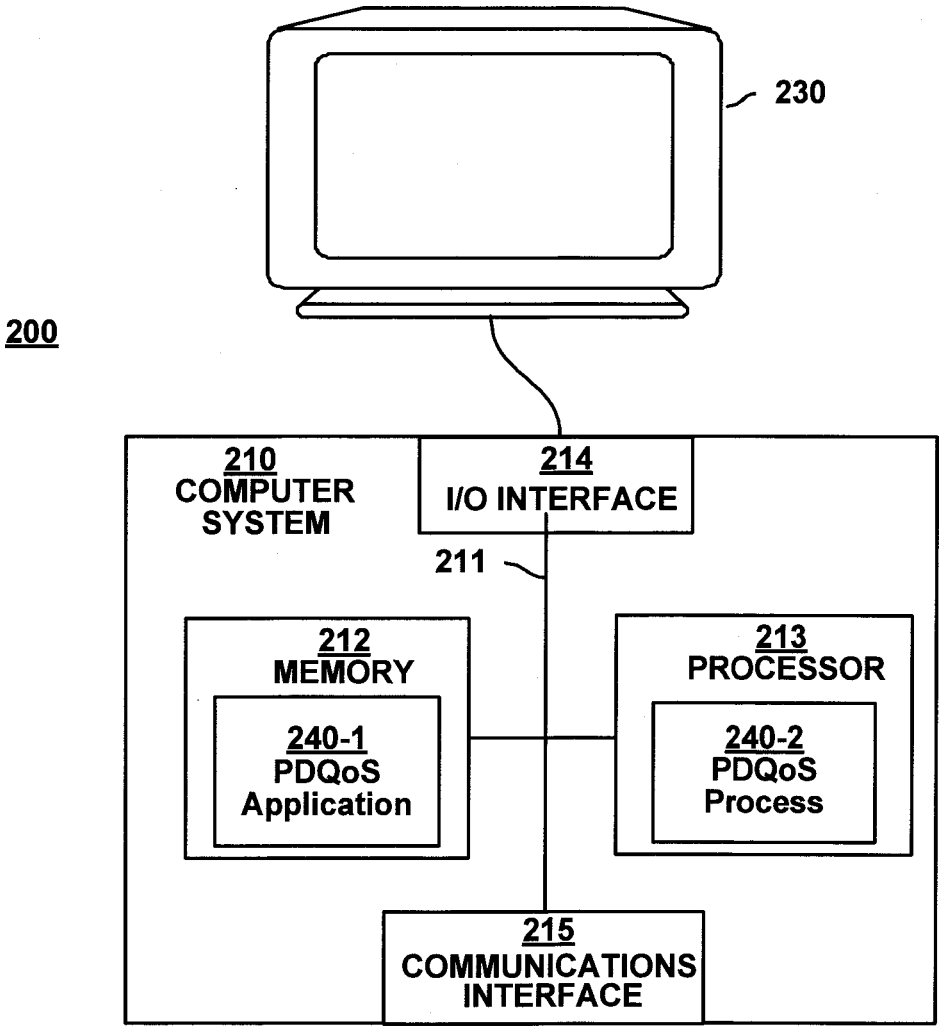

230
200
210
COMPUTER
SYSTEM
214
I/O INTERFACE
211
212
MEMORY
240-1
PDQoS
Application
213
PROCESSOR
240-2
PDQoS
Process
215
COMMUNICATIONS
INTERFACE

Figure 12

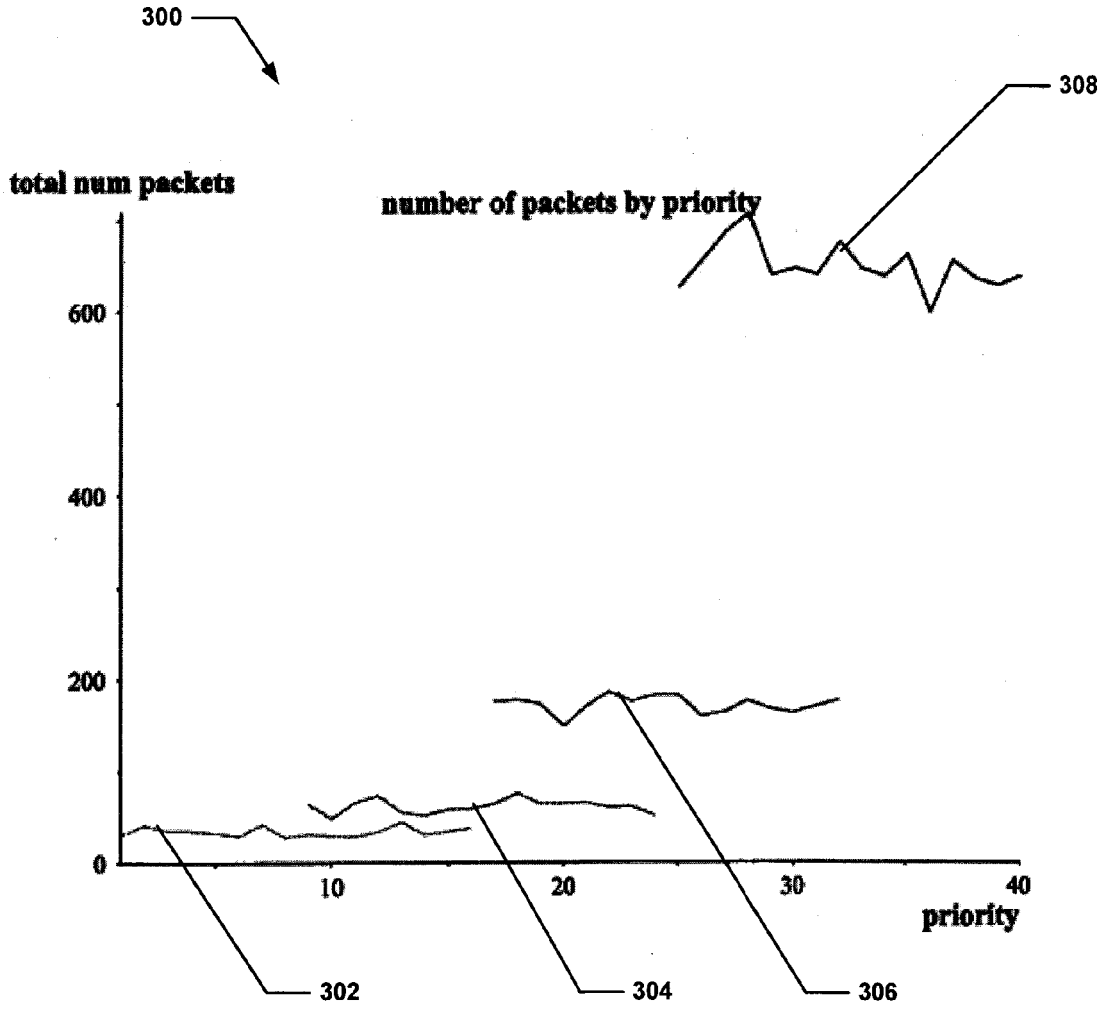

300
308
total num packets
number of packets by priority
600
400
200
0
10
20
30
40
priority
302
304
306

Figure 13

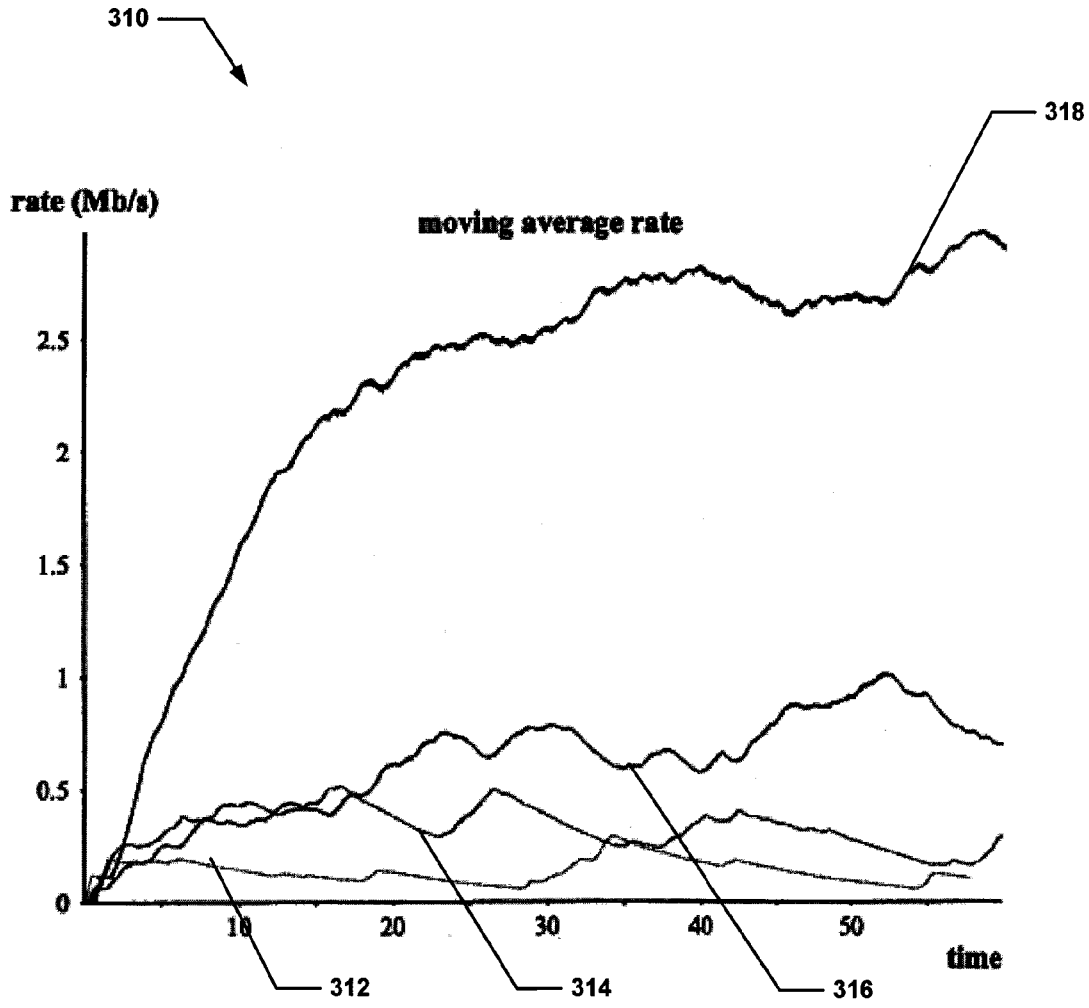

310
318
rate (Mb/s)
moving average rate
2.5
2
1.5
1
0.5
0
10
20
30
40
50
time
312
314
316

Figure 14

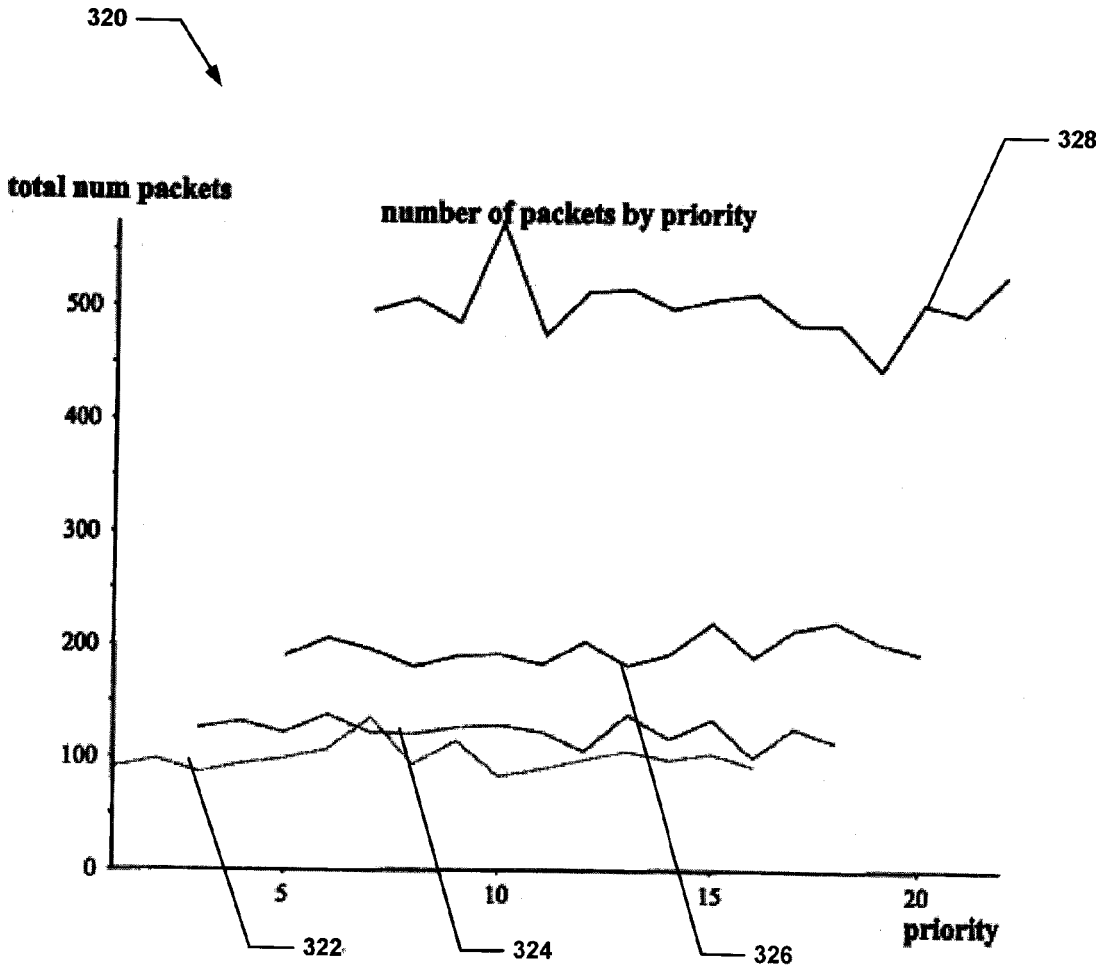

320
328
total num packets
number of packets by priority
500
400
300
200
100
0
5
10
15
20
priority
322
324
326

Figure 15

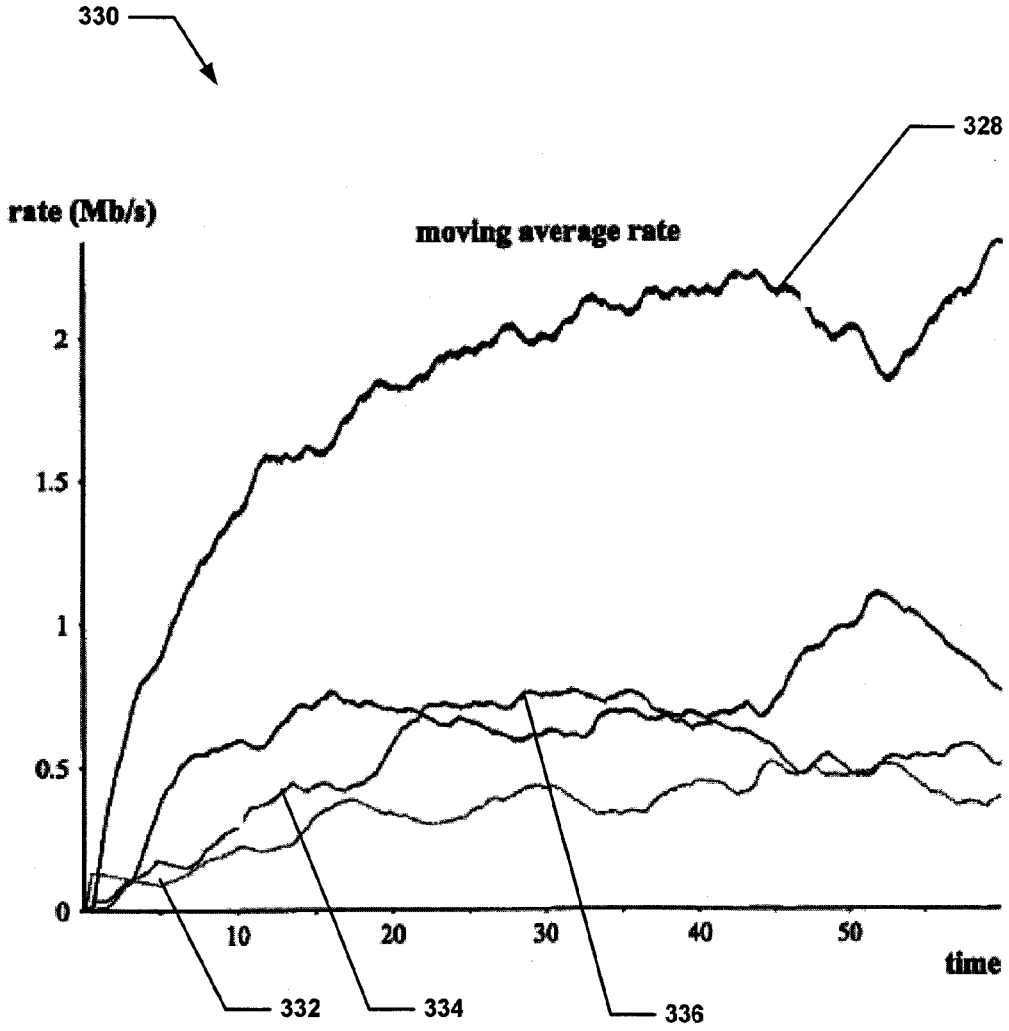

330
328
rate (Mb/s)
moving average rate
2
1.5
1
0.5
0
10
20
30
40
50
time
332
334
336

Figure 16

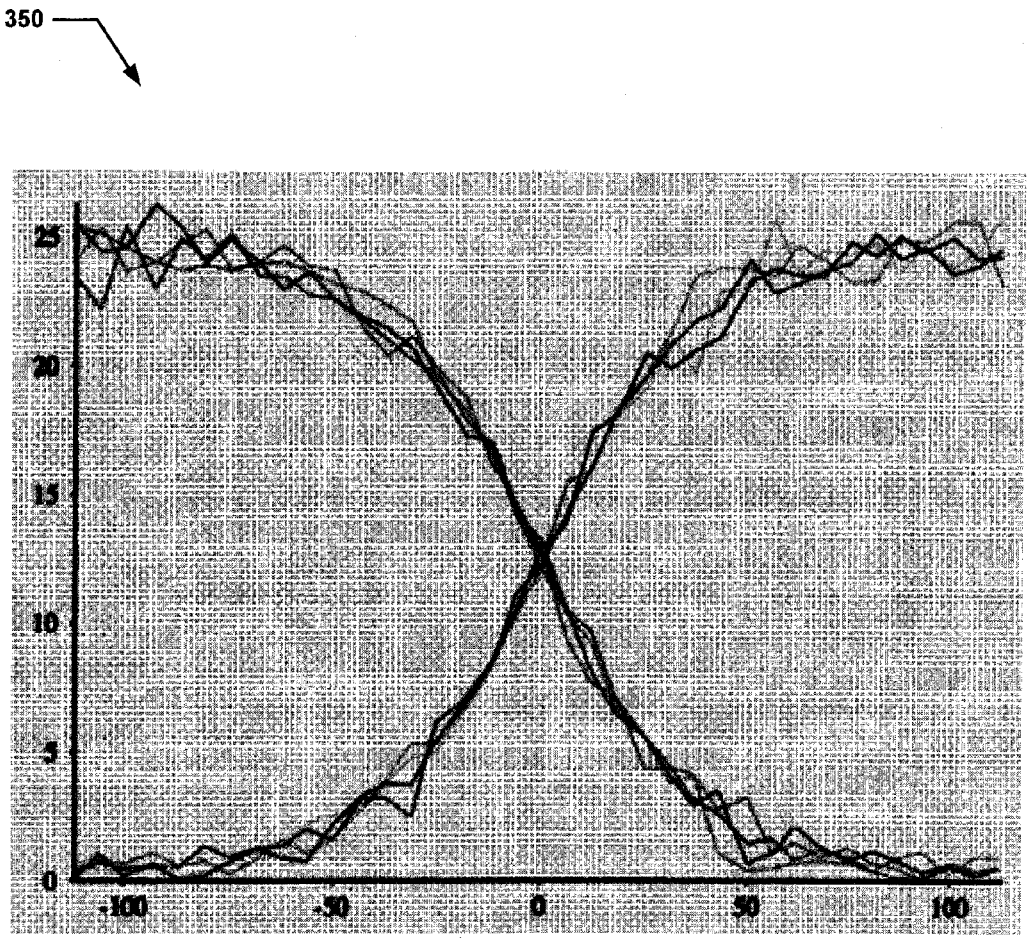


Figure 17

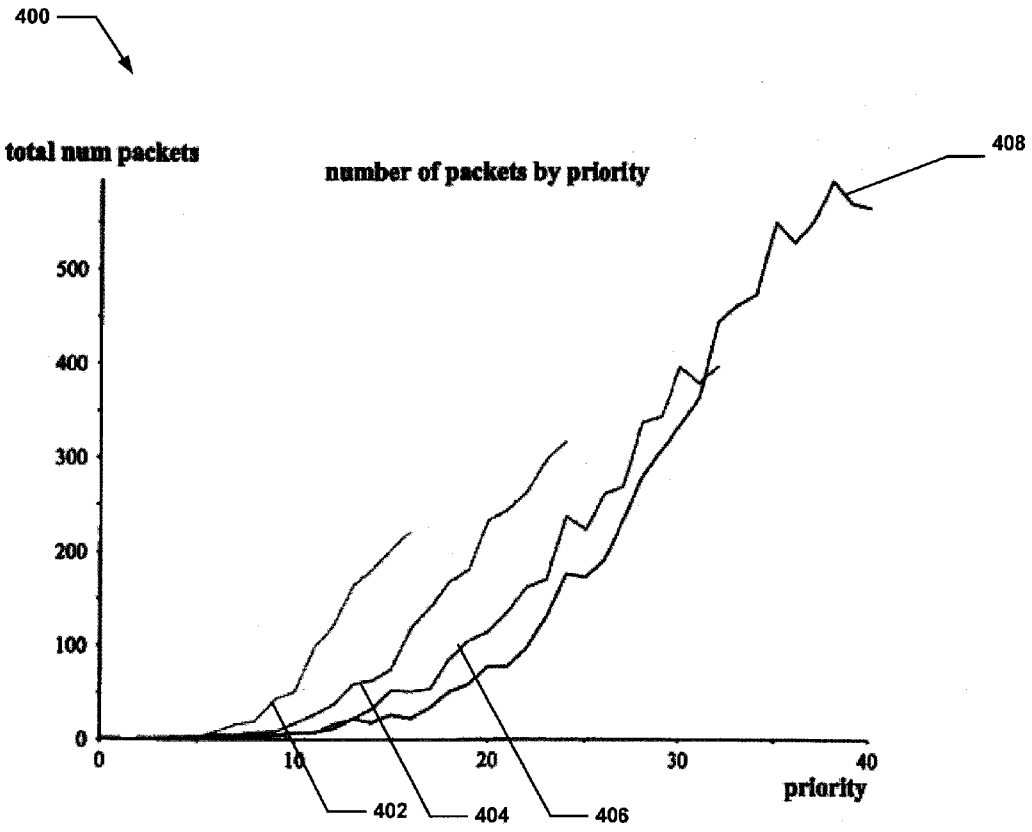

400
total num packets
number of packets by priority
408
500
400
300
200
100
0
0
10
20
30
40
priority
402
404
406

Figure 18

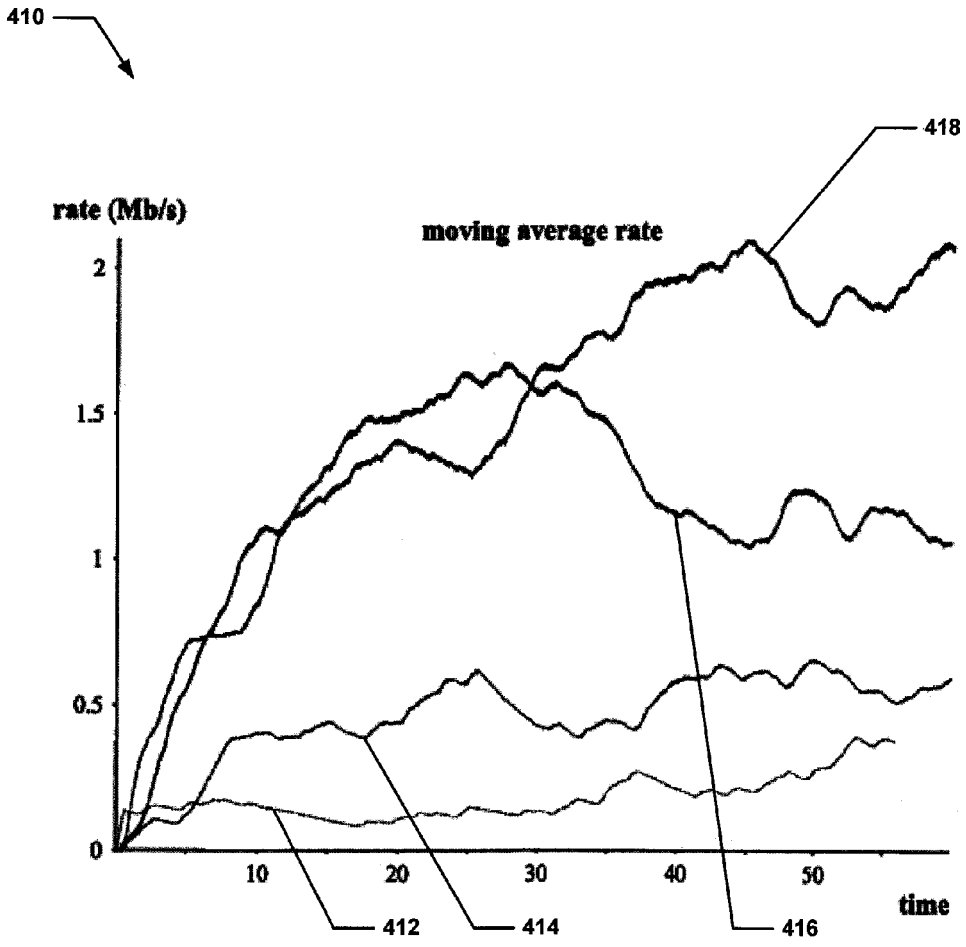

410
418
rate (Mb/s)
moving average rate
2
1.5
1
0.5
0
10
20
30
40
50
time
412
414
416

Figure 19

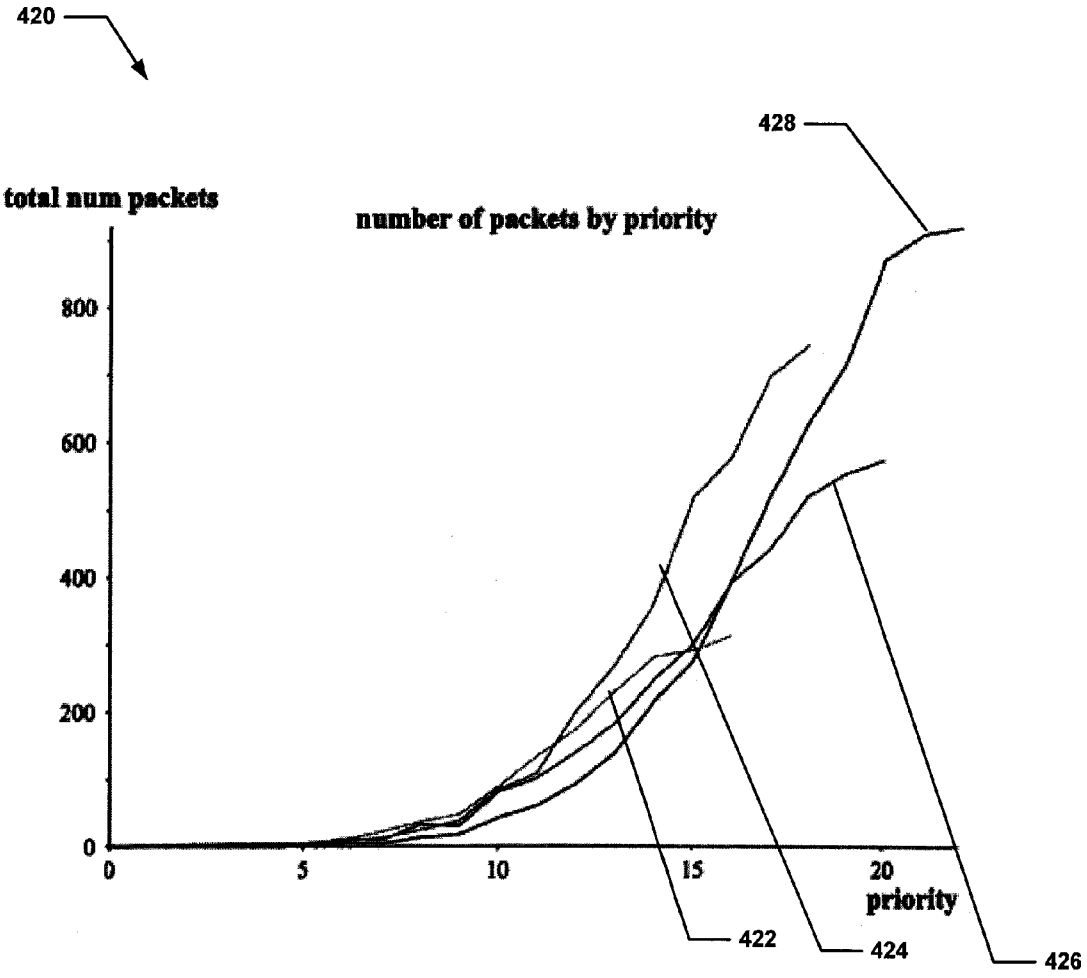

420
428
total num packets
number of packets by priority
800
600
400
200
0
0
5
10
15
20
priority
422
424
426

Figure 20

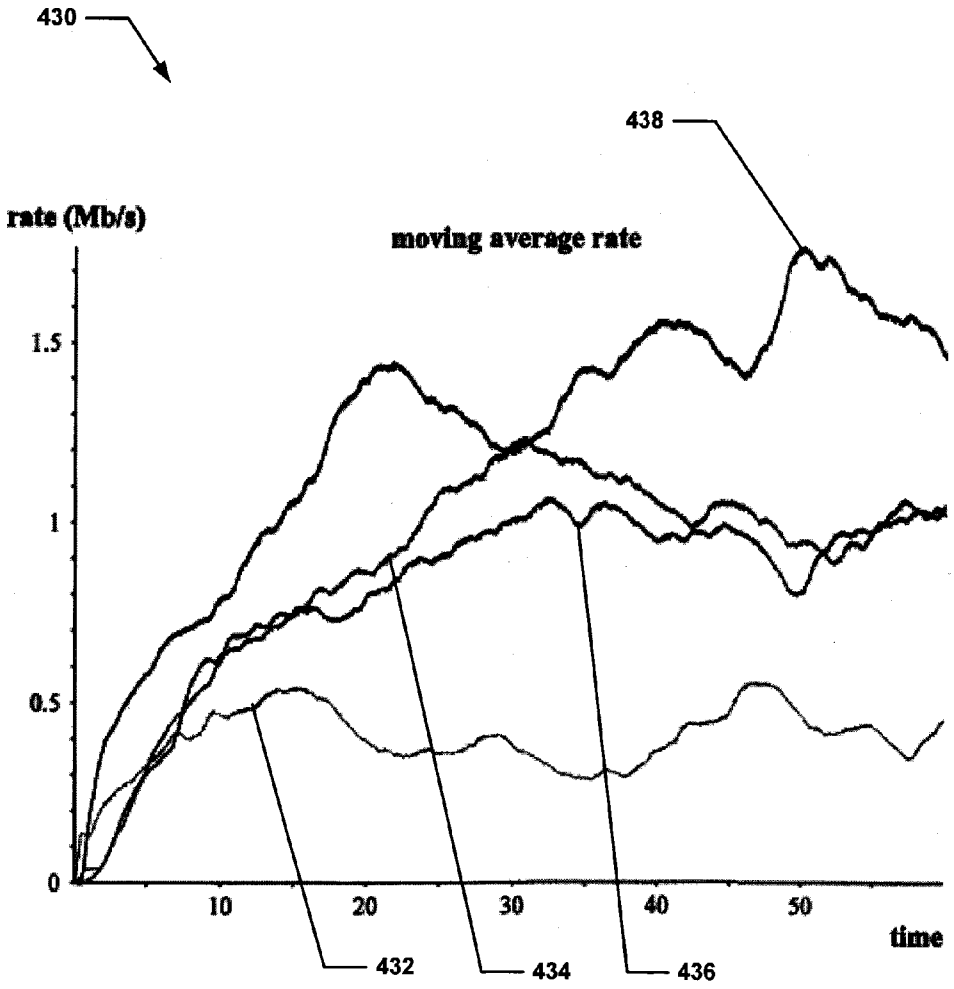

430
438
rate (Mb/s)
moving average rate
1.5
1
0.5
0
10
20
30
40
50
time
432
434
436

Figure 21

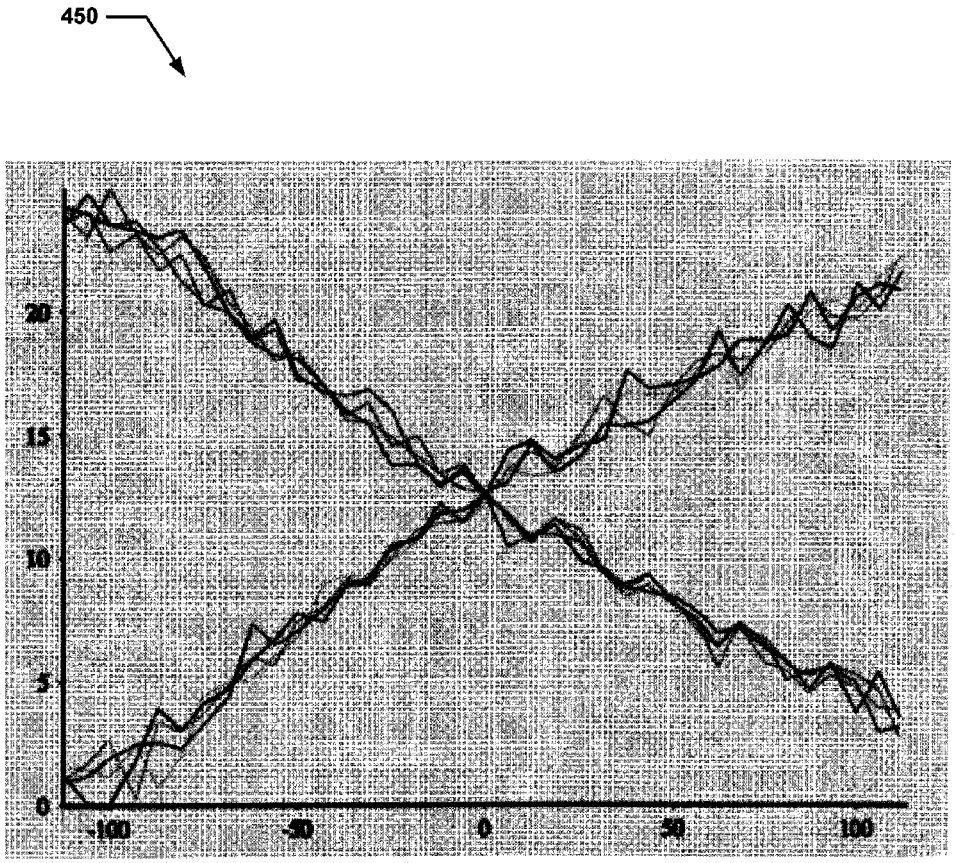


Figure 22

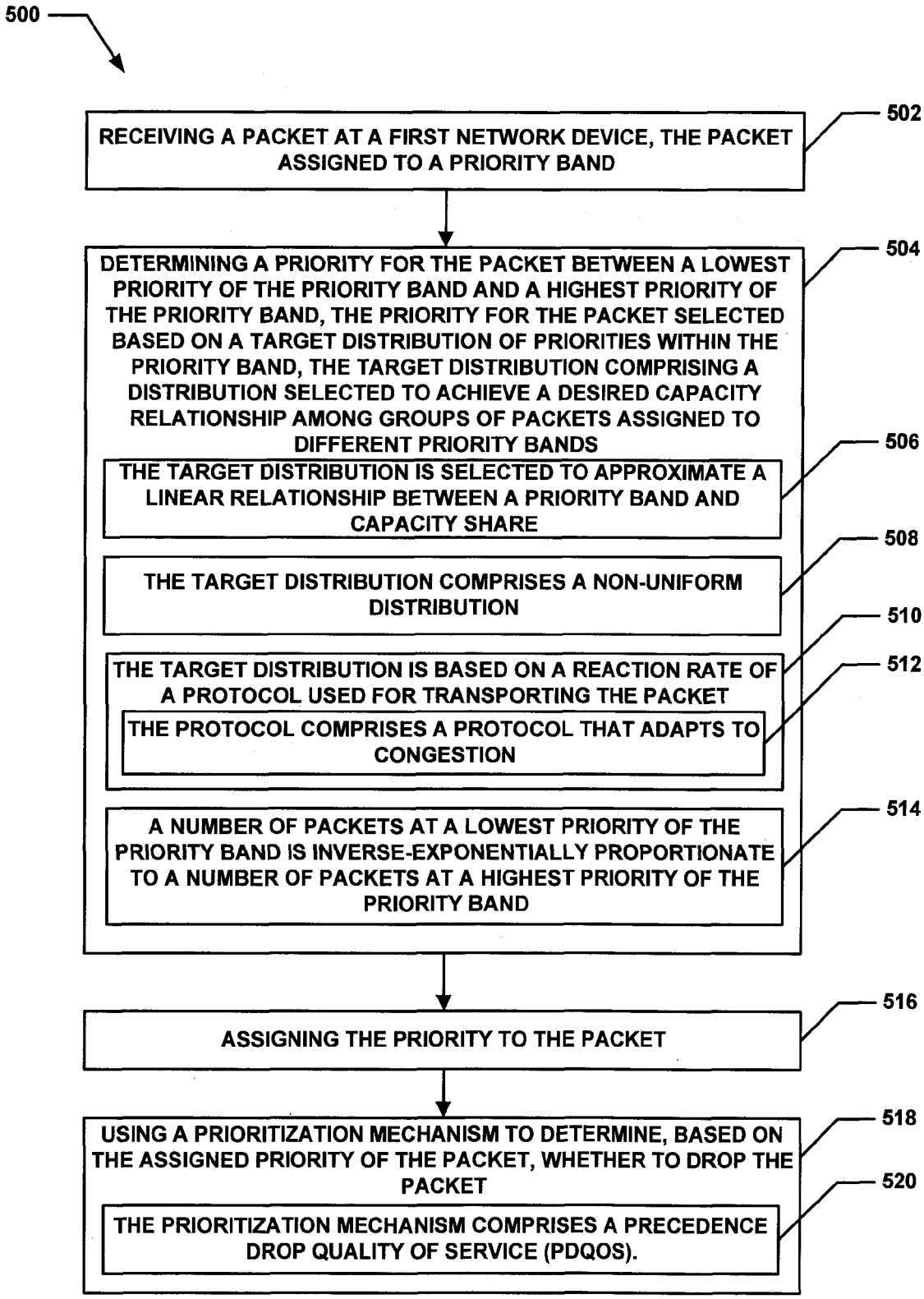

500
RECEIVING A PACKET AT A FIRST NETWORK DEVICE, THE PACKET ASSIGNED TO A PRIORITY BAND
502
DETERMINING A PRIORITY FOR THE PACKET BETWEEN A LOWEST PRIORITY OF THE PRIORITY BAND AND A HIGHEST PRIORITY OF THE PRIORITY BAND, THE PRIORITY FOR THE PACKET SELECTED BASED ON A TARGET DISTRIBUTION OF PRIORITIES WITHIN THE PRIORITY BAND, THE TARGET DISTRIBUTION COMPRISING A DISTRIBUTION SELECTED TO ACHIEVE A DESIRED CAPACITY RELATIONSHIP AMONG GROUPS OF PACKETS ASSIGNED TO DIFFERENT PRIORITY BANDS
504
THE TARGET DISTRIBUTION IS SELECTED TO APPROXIMATE A LINEAR RELATIONSHIP BETWEEN A PRIORITY BAND AND CAPACITY SHARE
506
THE TARGET DISTRIBUTION COMPRISES A NON-UNIFORM DISTRIBUTION
508
THE TARGET DISTRIBUTION IS BASED ON A REACTION RATE OF A PROTOCOL USED FOR TRANSPORTING THE PACKET
510
THE PROTOCOL COMPRISES A PROTOCOL THAT ADAPTS TO CONGESTION
512
A NUMBER OF PACKETS AT A LOWEST PRIORITY OF THE PRIORITY BAND IS INVERSE-EXPONENTIALLY PROPORTIONATE TO A NUMBER OF PACKETS AT A HIGHEST PRIORITY OF THE PRIORITY BAND
514
ASSIGNING THE PRIORITY TO THE PACKET
516
USING A PRIORITIZATION MECHANISM TO DETERMINE, BASED ON THE ASSIGNED PRIORITY OF THE PACKET, WHETHER TO DROP THE PACKET
518
THE PRIORITIZATION MECHANISM COMPRISES A PRECEDENCE DROP QUALITY OF SERVICE (PDQOS).
520

Figure 23

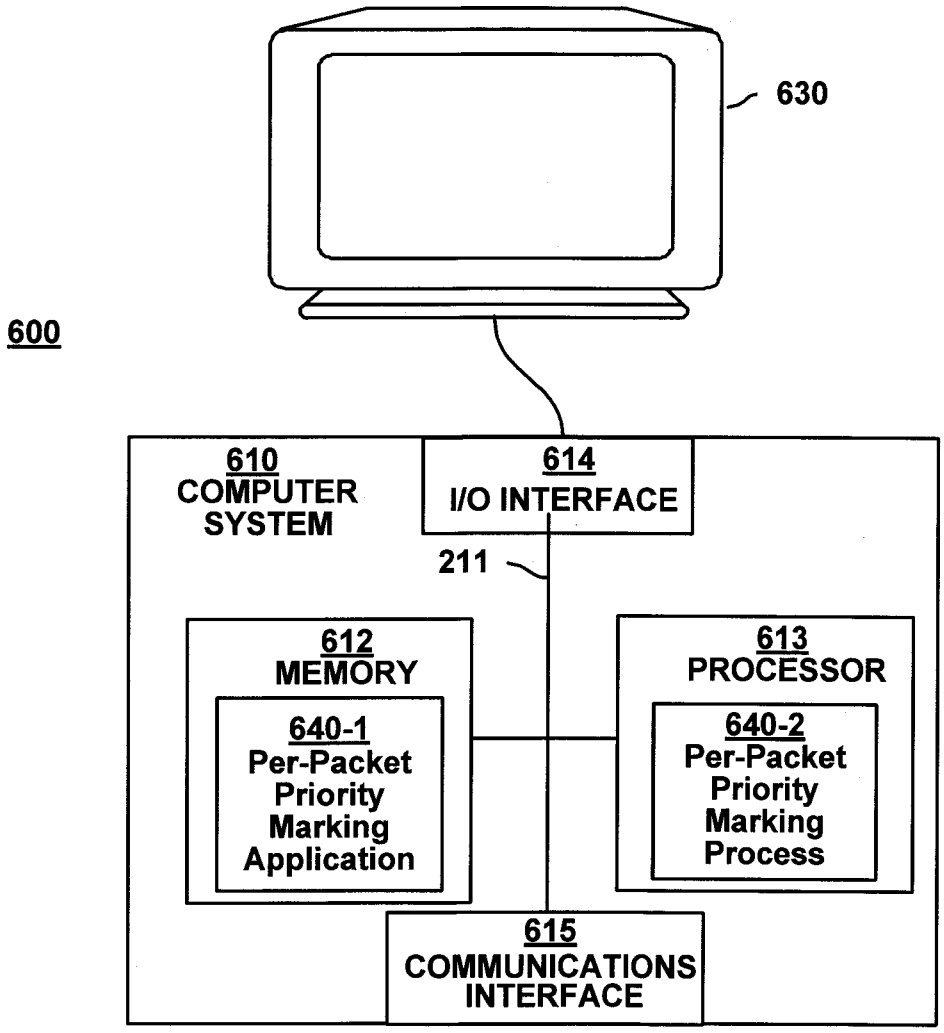

630
600
610
COMPUTER
SYSTEM
614
I/O INTERFACE
211
612
MEMORY
640-1
Per-Packet
Priority
Marking
Application
613
PROCESSOR
640-2
Per-Packet
Priority
Marking
Process
615
COMMUNICATIONS
INTERFACE

Figure 24

NON-UNIFORM PER-PACKET PRIORITY MARKER FOR USE WITH ADAPTIVE PROTOCOLS

CROSS-REFERENCE TO RELATED APPLICATIONS

This application is a continuation-in-part of U.S. application Ser. No. 12/200,264, filed Aug. 28, 2008, now U.S. Pat. No. 8,203,956.

GOVERNMENT RIGHTS

This invention was made with government support under Contract Number N66001-09-C-2073 awarded by DARPA. The government has certain rights in this invention.

BACKGROUND

A typical data communications network includes multiple host computers that are linked together by a combination of data communications devices and transmission media. In general, the host computers communicate by packaging data using a standardized protocol or format such as a network packet or cell (hereinafter generally referred to as a packet), and exchanging the packaged data through the data communications devices and transmission media.

In the field of computer networking and other packet-switched telecommunication networks, the traffic engineering term quality of service (QoS) refers to control mechanisms for achieving a desired service quality. Quality of service is the ability to provide different priority to different applications, users, or data flows, or to guarantee a certain level of performance to a data flow. For example, a required bit rate, delay, jitter, packet dropping probability and/or bit error rate may be guaranteed. Quality of service guarantees are important if the network capacity is insufficient, especially for real-time streaming multimedia applications such as Voice over IP (VoIP) and IP-TV, since these often require fixed bit rate and are delay sensitive, and in networks where the capacity is a limited resource, for example in cellular data communication.

For example, a distinction may be drawn between packets carrying video data (i.e., video packets belonging to a video QoS class) and packets carrying general data (i.e., general data packets belonging to a general data QoS class such as Best Effort Service). In this arrangement, a data communications device processes video packets through a network differently than general data packets due to different link resource availability and resources being allocated differently based on the QoS class of the packets.

There are different types of QoS transmission processing techniques. In one QoS transmission processing technique (hereinafter called QoS class-prioritized processing), a data communications device internally prioritizes the processing of different QoS class packets in accordance with a pre-established QoS policy. For example, in accordance with one such QoS policy, a data communications device gives higher priority to video packets relative to general data packets. Accordingly, if the data communications device simultaneously receives a video packet and a general data packet (e.g., through multiple input ports), the QoS policy directs the device to process the video packet before the general data packet. As a result, in QoS class-prioritized processing, packet destinations (i.e., receiving host computers) generally perceive different responses, or Qualities of Service, for different QoS classes (e.g., faster video transmissions than general data transmissions).

In another QoS processing technique (hereinafter called load-directed routing), the data communications device generally directs packets through the network based on network load conditions. For example, suppose that the data communications device has multiple output ports to data paths that eventually lead to the same remote destination (i.e., the same receiving host computer). When the data communications device receives a video packet destined for that remote destination, the device transmits the video packet out the output port expected to lead to a low traffic area of the network. On the other hand, when the device receives a general data packet destined for the same remote destination, the device transmits the general data packet out a different output port expected to lead to a high traffic area of the network. With this arrangement, the video packet should travel through the network faster than the general data packet. Accordingly, the load-directed routing technique generally provides the same result as the QoS class-prioritized routing technique. That is, packet destinations (i.e., receiving host computers) generally perceive different Quality of Service or responses for different QoS classes (e.g., quicker video transmissions than general data transmissions).

A network or protocol that supports QoS may agree on a traffic contract with the application software and reserve capacity in the network nodes, for example during a session establishment phase. During the session it may monitor the achieved level of performance, for example the data rate and delay, and dynamically control scheduling priorities in the network nodes. It may release the reserved capacity during a tear down phase.

A best-effort network or service does not support quality of service. An alternative to complex QoS control mechanisms is to provide high quality communication over a best-effort network by over-provisioning the capacity so that it is sufficient for the expected peak traffic load.

Another processing technique involves splitting traffic among multiple output queues which are then scheduled in some fashion (often using an algorithm called Weighted Fair Queuing (WFQ), which provides a specified minimum fraction of the link bandwidth to each queue feeding the link). Diffserv services (described below) often uses WFQ between traffic classes, with each class having its own queue.

Random early detection (RED), also known as random early discard or random early drop is an active queue management technique. It is also a congestion avoidance algorithm. In the traditional tail drop algorithm, a router or other network component buffers as many packets as it can, and simply drops the ones it cannot buffer. If buffers are constantly full, the network is congested. Tail drop distributes buffer space unfairly among traffic flows. Tail drop can also lead to Transmission Control Protocol (TCP) global synchronization as all TCP connections "hold back" simultaneously, and then step forward simultaneously. Networks become under-utilized and flooded by turns. RED addresses these issues by monitoring the average queue size and drops (or marks when used in conjunction with ECN) packets based on statistical probabilities. If the buffer is almost empty, all incoming packets are accepted. As the queue grows, the probability for dropping an incoming packet grows too. As the queue becomes full, drops become very likely, preventing the buffer from overflowing. RED is considered more fair than tail drop. The more a host transmits, the more likely it is that its packets are dropped. Early detection helps avoid global synchronization.

Per-flow QoS is also known as "intserv", for the IETF working group that defined it. It typically used WFQ with a single queue for each end-user traffic flow (e.g. a single TCP connection, or all traffic between a specific pair of sites).

Another type of management technique is knows as Differentiated Services (diffserv). In diffserv, users pay for the desired bandwidth speed and allotment (collectively referred to as a service level agreement (SLA)) for a packet flow. A flow is marked at every router, the marking indicating the level of service the packet flow is receiving. For example, a green marking indicates that the packet flow is within the SLA, a yellow marking indicates that the flow is slightly over the paid for SLA (but not by a large amount), a red indicator means the packet flow is over its SLA. A flow may transition between different markings as it traverses a network. Yet another type of management technique is known as Multi-Level Priority Marking (MLPM). In MLPM the IP Precedence field of a packet is used to encode the value of packets in an encoded video stream, so that the least valuable packets are dropped first.

SUMMARY

Conventional mechanisms such as those explained above suffer from a variety of deficiencies. One such deficiency is that conventional QoS implementations, for example Per-flow QoS, requires complex router state and signaling as well as a large number of queues which result in high equipment costs to implement and maintain. Per-flow QoS also requires high management complexity which also is costly, and as such Internet Service Providers (ISPs) typically do not use it. For traffic class QoS (e.g., diffserv), the cost and management is much simpler since there is no signaling and relatively few queues, but it really is just a more expensive best-effort service, therefore users tend not to purchase it since they are not guaranteed the performance they desire. In MLPM, because the precedence levels were assigned to different queues in the routers, the packets would arrive wildly out-of-order. Reordering the packets into their correct order is difficult and expensive, and become even more so at higher speeds.

In existing QoS enforcement, packet ordering may be changed to favor "important" flows or to rate-shape flows or classes, as well as to keep links busy if any traffic is available. The ordering method may be by priority or weighted fair queuing. As stated earlier, reordering the packets into their correct order is difficult and expensive, and become even more so at higher speeds. Packets are dropped if queues back up too far or if Random Early Discard (RED) triggers (to rate control TCP without synchronized misbehavior).

Embodiments of the invention significantly overcome such deficiencies and provide mechanisms and techniques that provide precedence drop quality of service (PDQoS) methodology. The precedence drop QoS methodology is simple to configure and manage, offers several advantages to end uses, and is straightforward to implement.

The precedence drop QoS methodology assigns a precedence drop value to each packet and packets are dropped if a sum of queued packet sizes of all packets having a same or higher drop precedence value than the newly received packet is larger than a threshold value. This provides user control over congestion and removes the need for packet re-ordering for QoS enforcement. Thus QoS enforcement can be done entirely in a simple packet-drop decision, without employing multiple output queues and a complex multi-queue scheduler. Precedence-based dropping can also replace RED, and is of similar (low) complexity. (In actual fact, it's easier, because RED normally require doing integer division for every packet, PDQoS only needs to do much simpler adds and subtracts. However, it does need a small state table, which RED implementations don't require.)

Different packets in same flow may have radically different drop precedences (i.e. value to user).

In a particular embodiment of a method for providing drop precedence quality of service, the method includes determining a drop precedence value for a packet and inserting the drop precedence value into the packet. The method further includes transmitting the packet having the drop precedence value inserted therein. Typically this is accomplished at a grooming router or by the originating host.

In another embodiment of a method of providing drop precedence quality of service, the method includes receiving a plurality of packets, at least one packet of the plurality of packets having a drop precedence value associated therewith. The method further includes determining for a newly received packet whether a sum of queued packet sizes of all packets having a same or higher drop precedence value than the newly received packet is larger than a threshold value. The method additionally includes dropping the newly received packet when the sum of queued packet sizes of all packets having a same or higher drop precedence value than the newly received packet is larger than the threshold value.

Other embodiments include a computer readable medium having computer readable code thereon for providing drop precedence quality of service. The computer readable medium includes instructions for determining a drop precedence value for a packet and instructions for inserting the drop precedence value into the packet. The computer readable medium further includes instructions for transmitting the packet having the drop precedence value inserted therein.

In another embodiment, a computer readable medium having computer readable code thereon for providing drop precedence quality of service includes instructions for receiving a plurality of packets, at least one packet of the plurality of packets having a drop precedence value associated therewith. The computer readable medium further includes instructions for determining for a newly received packet whether a sum of queued packet sizes of all packets having a same or higher drop precedence value than the newly received packet is larger than a threshold value. The computer readable medium additional includes instructions for dropping the newly received packet when the sum of queued packet sizes of all packets having a same or higher drop precedence value than the newly received packet is larger than the threshold value.

In still another embodiment, a particular method for non-uniform per-packet priority marking for use with adaptive protocols includes receiving a packet at a first network device, the packet assigned to a priority band. The method further includes determining a priority for the packet between a lowest priority of the priority band and a highest priority of the priority band, the priority for the packet selected based on a target distribution of priorities within the priority band, the target distribution comprising a distribution selected to achieve a desired capacity relationship among groups of packets assigned to different priority bands. Additionally, the method includes assigning the selected priority to the packet.

In yet still another embodiment, a computer readable medium having computer readable code thereon for non-uniform per-packet priority marking for use with adaptive protocols includes instructions for receiving a packet at a first network device, the packet assigned to a priority band. The computer readable medium further includes determining a priority for the packet between a lowest priority of the priority band and a highest priority of the priority band, the priority for the packet selected based on a target distribution of pri-

orities within the priority band, the target distribution comprising a distribution selected to achieve a desired capacity relationship among groups of packets assigned to different priority bands. Additionally, the computer readable medium includes instructions for assigning the selected priority to the packet.

Still other embodiments include a computerized device, configured to process all the method operations disclosed herein as embodiments of the invention. In such embodiments, the computerized device includes a memory system, a processor, communications interface in an interconnection mechanism connecting these components. The memory system is encoded with a process that provides for non-uniform per-packet priority marking for use with adaptive protocols as explained herein that when performed (e.g. when executing) on the processor, operates as explained herein within the computerized device to perform all of the method embodiments and operations explained herein as embodiments of the invention. Thus any computerized device that performs or is programmed to perform the processing explained herein is an embodiment of the invention.

Other arrangements of embodiments of the invention that are disclosed herein include software programs to perform the method embodiment steps and operations summarized above and disclosed in detail below. More particularly, a computer program product is one embodiment that has a computer-readable medium including computer program logic encoded thereon that when performed in a computerized device provides associated operations for non-uniform per-packet priority marking for use with adaptive protocols as explained herein. The computer program logic, when executed on at least one processor with a computing system, causes the processor to perform the operations (e.g., the methods) indicated herein as embodiments of the invention. Such arrangements of the invention are typically provided as software, code and/or other data structures arranged or encoded on a computer readable medium such as an optical medium (e.g., CD-ROM), floppy or hard disk or other a medium such as firmware or microcode in one or more ROM or RAM or PROM chips or as an Application Specific Integrated Circuit (ASIC) or as downloadable software images in one or more modules, shared libraries, etc. The software or firmware or other such configurations can be installed onto a computerized device to cause one or more processors in the computerized device to perform the techniques explained herein as embodiments of the invention. Software processes that operate in a collection of computerized devices, such as in a group of data communications devices or other entities can also provide the system of the invention. The system of the invention can be distributed between many software processes on several data communications devices, or all processes could run on a small set of dedicated computers, or on one computer alone.

It is to be understood that the embodiments of the invention can be embodied strictly as a software program, as software and hardware, or as hardware and/or circuitry alone, such as within a data communications device. The features of the invention, as explained herein, may be employed in data communications devices and/or software systems for such devices.

Note that each of the different features, techniques, configurations, etc. discussed in this disclosure can be executed independently or in combination. Accordingly, the present invention can be embodied and viewed in many different ways.

Also, note that this summary section herein does not specify every embodiment and/or incrementally novel aspect of the present disclosure or claimed invention. Instead, this summary only provides a preliminary discussion of different embodiments and corresponding points of novelty over conventional techniques. For additional details, elements, and/or possible perspectives (permutations) of the invention, the reader is directed to the Detailed Description section and corresponding figures of the present disclosure as further discussed below.

BRIEF DESCRIPTION OF THE DRAWINGS

The foregoing will be apparent from the following more particular description of preferred embodiments of the invention, as illustrated in the accompanying drawings in which like reference characters refer to the same parts throughout the different views. The drawings are not necessarily to scale, emphasis instead being placed upon illustrating the principles of the invention.

FIG. **1** depicts a graph of drop precedence value versus a queue length of higher-residence packets;

FIG. **2** depicts a graph of drop precedence value versus time for TCP packet flows;

FIG. **3** depicts a graph of drop precedence value versus time for video packet flows;

FIG. **4** depicts a graph of drop precedence value versus a queue length of higher-precedence packets for a two-level priority scheme;

FIG. **5** depicts a graph of drop precedence value versus a packet rate for a Service Level Agreement (SLA);

FIG. **6** depicts a graph of drop precedence value versus a packet rate for a preferred user Service Level Agreements (SLA);

FIG. **7** depicts a graph of drop precedence value versus a packet rate per level for a three tier user class Service Level Agreements (SLA);

FIG. **8** depicts a graph of drop precedence value versus a packet rate per level for a three tier pre-emption-based Service Level Agreements (SLA);

FIG. **9** depicts a graph of drop precedence value versus a packet rate for different SLA users;

FIGS. **10**A and **10**B are a flow diagram of a particular embodiment of a method for performing drop precedence quality of service in accordance with embodiment of the present invention;

FIG. **11** is a flow diagram of a particular embodiment of a method for assigning drop precedence values for packets in accordance with embodiment of the present invention;

FIG. **12** illustrates an example computer system architecture for a computer system that performs drop precedence quality of service in accordance with embodiments of the invention;

FIG. **13** illustrates four TCP flows using uniform distribution priority assignment showing a number of packets per priority;

FIG. **14** illustrates four TCP flows using uniform distribution priority assignment showing a number of throughput over time;

FIG. **15** illustrates four TCP flows using uniform distribution priority assignment showing a number of packets per priority;

FIG. **16** illustrates four TCP flows using uniform distribution priority assignment showing a number of throughput over time;

FIG. **17** illustrates a relationship between an amount of priority overlap between flows and a percentage of bottleneck capacity used by each flow;

FIG. **18** illustrates four TCP flows using normal distribution priority assignment showing a number of packets per priority in accordance with embodiments of the invention;

FIG. **19** illustrates four TCP flows using uniform distribution priority assignment showing a number of throughput over time in accordance with embodiments of the invention;

FIG. **20** illustrates four TCP flows using uniform distribution priority assignment showing a number of packets per priority in accordance with embodiments of the invention;

FIG. **21** illustrates four TCP flows using uniform distribution priority assignment showing a number of throughput over time in accordance with embodiments of the invention;

FIG. **22** illustrates a relationship between an amount of priority overlap between flows and a percentage of bottleneck capacity used by each flow in accordance with embodiments of the invention;

FIG. **23** is a flow diagram of a particular embodiment of a method for non-uniform per-packet priority marking for use with adaptive protocols in accordance with embodiment of the present invention; and

FIG. **24** illustrates an example computer system architecture for a computer system that performs non-uniform per-packet priority marking for use with adaptive protocols in accordance with embodiments of the invention.

DETAILED DESCRIPTION

A method and apparatus are described for providing drop precedence quality of service (PDQoS). A grooming router, source host computer or application is used wherein a drop precedence value for a packet is determined and inserted into the packet. The packet having the drop precedence value inserted therein is then transmitted. A router receives a plurality of packets, at least one packet of the plurality of packets having a drop precedence value associated therewith. The router then determines for a newly received packet having a drop precedence value associated therewith whether a sum of queued packet sizes of all packets having a same or higher drop precedence value than the newly received packet is larger than a threshold value. The router will drop the newly received packet when the sum of queued packet sizes of all packets having a same or higher drop precedence value than the newly received packet is larger than the threshold value, or forward the packet when the sum of queued packet sizes of all packets having a same or higher drop precedence value than the newly received packet are not larger than the threshold value. Only packet drop behavior is used to provide a QoS type effect, as well as providing user control over congestion and eliminating the need for packet re-ordering.

Referring to FIG. **1**, a graph **10** is shown. In graph **10**, the vertical axis is the drop precedence. Here, 256 precedence levels are possible, by use of an 8-bit drop precedence field. The horizontal axis shows the sum of the queue length of higher precedence packets. Also shown is a "Drop region" **12**, defined by the threshold value L_{drop} wherein packets falling into this region are dropped since the sum of queued packet sizes of all higher drop precedence values is greater than this latency threshold value. When the queue exceeds the L_{drop} value for a particular precedence, any packets received having that precedence value will be dropped. The line P_{cutoff} shows the intersection of curve **14** with the L_{drop} value.

In one particular embodiment a "grooming" enterprise border router is used to mark the packets with a drop precedence value (other embodiments may use a source host computer or application for this task). The content and/or classification of the packet is looked at, and a classification is performed based on a predetermined policy. The grooming router can use almost the same classification rules as are currently used for classifying flows by enterprise border routers. The difference is that the classifying is done on a per packet basis instead of a per flow basis. Stated differently, the difference is that in addition to determining a flow type by address, protocol and port information, the marking process can use additional information from the packet to choose particular values for specific packets within the flow. There are several different ways for the grooming router to assign a drop precedence value to a packet. Some packets in a same flow are more important than other packets of the same flow, and the more important packets are given a higher drop precedence value. For example, in a video application, key frames would be given a higher drop precedence value than incremental frames. While this can be done at the grooming router, this may also be done at the application server itself, and the grooming router would not be required to mark these packets with a drop precedence value since the packets have already been classified.

Referring now to FIG. **2**, a Transmission Control Protocol (TCP) marking example is shown. A graph **20** is shown having the drop precedence value on a vertical axis and time on the horizontal axis. The drop precedence of control packets (e.g. Open **22** or Close **24** or acknowledgements **26**) is given a higher drop precedence value than other packets (data **28**). Since the control packets (**22**, **24**, **26**) are more important than the data packets **28** the control packets are given a higher drop precedence value, whereas the data packets are given various lower drop precedence values. For example, if a user is accessing a web site it is more important to establish the link (using the control packets) since the user doesn't want to wait a period of time to open a web page, than it is for a particular file to download (the data packets) from the web page.

The data packets are given random or pseudo-random drop precedence values from a range of values that the classifier assigned to the TCP session. This randomness, in conjunction with a PDQoS enforcement mechanism at congestion points, performs the same function as RED does in current routers—it causes flows to be penalized semi-randomly, based on queue depths, causing TCP backoff to be de-synchronized at a given congestion point. (It may also be worth noting that PDQoS-marked flows will still work with classical RED-based routers, if the classical router classifies all the packets of various drop precedences used by the flow as belonging to the same queue.)

Referring now to FIG. **3**, a video-marking example is shown. In this graph **30**, the vertical axis is again the drop precedence value and the horizontal axis is time. Here, all packets of the entire video are given a relatively high drop precedence value, since it is desirable not to drop any packets of the video as the quality of the video would be degraded when packets are dropped. Within the video flow, packets which are key frames **32** are given the highest drop precedence value while incremental packets **34** of the video flow are given lower drop precedence values than the key frames **32**, but are still given a relatively high drop precedence value as compared to other data that might be occurring on the router. Thus, in this example, incremental video frames **34** would be dropped prior to key frames **32** being dropped in the event congestion is encountered.

The drop precedence methodology is straight forward to implement and manage. One big advantage offered by the drop precedence methodology is that since no packet re-ordering is done, no packet re-ordering has to be handled. As a direct result, a single simple, cheap Dynamic Random Access Memory (DRAM) First In/First Out (FIFO) buffer can be used. Additionally, there is no signaling or per-flow

state that has to be maintained or monitored. A per-port set of traffic counters (a total of 256, one for each drop precedence level) can be used. On a per customer arrangement, there are two per-customer Service Level Agreement (SLA) variables (P_{max}, BW_{total}). BW_{total} is the total bandwidth the customer is buying from the network provider. P_{max} is the maximum precedence value the network provider allows the user to send at. In this type of environment, the Internet Service Provider (ISP) is not responsible for managing the edge routers, instead the customers manage edge-grooming routers and define how their traffic is going to be treated by the network. This is one particular type of SLA, and probably the simplest (and thus most likely to be used). The BWtotal amount is split evenly among all the drop precedence values 0 . . . Pmax—i.e., the customer gets BWtotal/(Pmax+1) bandwidth available over each of the precedences 0 . . . Pmax, measured over whatever time interval the SLA is applied to.

There can be two types of network traffic. Traffic that is latency sensitive (low-latency traffic) such as voice and video conferencing, and traffic that is not latency sensitive (normal). Low latency support can be accomplished by providing a two-level priority scheme. The two-level priority scheme incorporates two output queues, a low-latency output queue and a normal output queue. Thus, for incoming traffic, traffic having a low-latency is sent to the low-latency queue and normal traffic is sent to the normal queue. The low-latency queue is high priority, and typically only a relatively few precedence levels are reserved for low-latency traffic. Low-latency traffic tends to be inelastic, and therefore doesn't require many levels. It is also possible to overlap levels if other means are used to pick a queue (e.g. which port number is used). The low-latency traffic uses a separate small SRAM buffer which has its own queue threshold (L_{lowlat}) which is lower than that of normal traffic.

An example of this is shown in FIG. **4**. Here a graph **40** is shown wherein the vertical axis is the drop precedence. Here, 256 precedence levels are possible, by use of an 8-bit drop precedence field. The horizontal axis shows the sum of the queue length of higher precedence packets. Also shown is a "Drop Region" **12**, defined by the threshold value L_{drop} wherein packets falling into this region are dropped since the sum of queued packet sizes of all higher drop precedence values is greater than this latency threshold value. When the queue exceeds the L_{drop} value for a particular precedence, any packets received having that precedence value will be dropped. The line P_{cutoff} shows the intersection of curve **14** with the L_{drop} value. In addition, for low-latency traffic a precedence level L_{lowlat} **40** is determined for this type of traffic. Also shown is a "low-latency drop region" **44**, defined by the threshold value L_{lowlat} wherein packets falling into this region are dropped since the sum of queued packet sizes of all higher drop precedence values is greater than this latency threshold value L_{lowlat}. When the queue exceeds the L_{lowlat} value for a particular precedence, any packets received having that precedence value will be dropped. The line P_{lowlat} **42** shows the first precedence value in the low-latency region (i.e., it defines the boundary between drop precedences used for low-latency traffic, and drop precedences used for normal traffic).

The presently disclosed methods and techniques for providing precedence drop QoS replaces RED for elastic TCP traffic, and gives applications and/or users control over what traffic is most important and what traffic is dropped first which in turn provides new capabilities and/or applications.

One such application involves call admission control for inelastic traffic (e.g. VoIP). If the precedence level is set to a specific value for all VoIP traffic, then the VoIP traffic is getting through entirely until there is enough congestion in the network that the traffic having a higher precedence than the VoIP traffic will cause the VoIP traffic to completely drop out, which his the desired behavior for VoIP traffic. This is compared to conventional techniques where only a fraction of the VoIP traffic (e.g., ten percent) drops out. However, for VoIP traffic, if ten percent is going to be dropped, then the whole VoIP traffic should be dropped.

Precedence drop QoS supports priority applications and pre-emption applications. The basic mechanism allows users to define customized QoS behaviors without the ISP being required to define the customized behavior. PDQoS statistics collection is added at an edge of a network. Customer traffic is then counted at each of 256 levels. For a fixed customer SLA, the total amount of traffic is limited per level (may also limit the levels usable). The network provider can either enforce the SLA and drop traffic exceeding the SLA or can charge additional fees for excess use. The network provider can also use a flexible pricing (usage-sensitive) arrangement, wherein the customer is charged a different price per 1 Mbit sent for each level. Customers could optimize marking to minimize costs. Such an environment would require the far end to feed back which levels get through, with the customers using the lowest precedence level that gets through to minimize their costs.

Referring now to FIG. **5**, a graph **50** is shown, the graph relating to a sample SLA agreement. In this example, most traffic is sent at a lower drop precedence level, and the higher precedence levels are used during bursts of traffic. FIG. **6** shows a graph **60** relating to an alternate SLA agreement. The difference between the SLAs shown in FIGS. **5** and **6** is that the SLA of FIG. **5** limits the normal priority traffic to a smaller drop precedence range than in FIG. **6**, though the total bandwidth for normal traffic (area of the lower rectangle) is the same for both customers. This allows the customer with the "premium" SLA shown in FIG. **6** to get traffic through congestion points that would totally stop traffic from a customer using the SLA shown in FIG. **5**, though only some fraction of his traffic would be getting through (above the Pcutoff value of the congestion point). Both SLAs include a small amount of low latency bandwidth, but the precedence ranges used for the two SLAs are different, again giving the customer with the SLA shown in FIG. **6** an advantage when congestion occurs.

FIG. **7** shows a graph **70** relating to a 3-tier user class SLA arrangement. In this environment, there are three distinct service levels, namely gold, silver and bronze. The gold level includes a higher top-end drop precedence value than the silver or bronze levels, as well as all drop precedence values beneath the top-end drop precedence value. The silver level includes a higher top-end drop precedence value than the bronze level, as well as all drop precedence values beneath the top-end drop precedence level. The silver level top-end drop precedence value is less than the gold level top-end drop precedence value. The bronze level includes a lower top-end drop precedence value than the silver or gold levels, as well as all drop precedence values beneath the top-end drop precedence value. FIG. **8** shows a graph **80** relating to a different 3-tier SLA, referred to herein as a pre-emption based SLA. In this type of arrangement, the gold level of service includes a set of drop precedence values that are higher than, and do not overlap with, either the silver or bronze service levels. Similarly, the silver level of service includes a set of precedence value that are less than the gold level set of precedence values but are higher than the bronze level of precedence values. The bronze level of service includes a set of drop precedence values that are lower than, and do not overlap with, either the silver or gold service levels.

FIG. **9** is a graph **90** showing how available bandwidth can be distributed. In this example, VoIP and Video Teleconferencing (VTC) having the highest priority. Mission-critical servers have the next highest priority followed by priority users then routine users. Outside users have the next to lowest priority, with elastic servers having the lowest priority. FIG. **9** illustrates how the classification rules in a grooming router might carve up the available bandwidth at various drop precedence levels. Note that the various normal-latency applications use all the precedence levels, but some of them get more of various higher drop precedences. (Again, the area of a block indicates its total bandwidth. Inverse-L-shaped blocks allow the high-precedence applications to use both a significant amount of higher-precedence bandwidth, as well as use some additional bandwidth at lower precedences, when those levels are getting through.) This is one of any number of such carve-up schemes; generally the end-user would define these as policies to the enterprise border grooming router. In fact, these policies might be defined using a simple GUI interface that presented a diagram much like FIG. **9**.

Another use of PDQoS utilizes feedback of Pcutoff to the source (or its grooming router). This would have either the destination or intermediate routers report the minimum value of Pcutoff along the route of the flow to the source, allowing the source to simply not send packets with lower drop precedence values. A variant of this has upstream routers being informed of Pcutoff values on each link of the network, and using this information to pre-emptively drop packets that will likely be dropped as the flow progresses downstream. This frees up bandwidth that would be normally be used by packets being dropped further along each flow's path through the network core, giving the network a higher effective capacity in congestion scenarios.

Still another use of PDQoS involves PDQoS-based Routing. If the link P_{cutoff} is sent via routing, traffic can be re-routed to avoid congestion (congestion-aware routing). The router would select a path to the destination with lowest max P_{cutoff} value. This is potentially better than a link utilization metric, as P_{cutoff} implies not just link load, but also value to user. Like any differential routing scheme, routing performance will be subject to stability and routing-loop issues.

The PDQoS enforcement algorithm works as follows. There are three sub-algorithms which typically operate as independent processes or threads: initialization, packet arrival processing, and packet transmission processing.

The initialization thread is performed once at the start of time (e.g. at interface reset). The pseudo code for this is shown below.

1) QLen[p] is set to 0 for all drop precedences p in the range 0 . . . Pmax

The packet arrival processing thread pseudo code is listed below.

1) Wait for a packet to arrive, and extract its drop precedence value P and packet length L
2) If QLen[P]+L>Ldrop, discard the packet and go to 1) to wait for another packet to arrive, else go to 3)
3) For all values p<=P (from the packet), QLen[p]=QLen [p]+L
4) Queue the packet on the end of the transmission queue
5) Go to step 1) to wait for another packet arrival

The packet transmission processing thread pseudo code is listed below.

1) Wait until the transmitter can accept another packet
2) Dequeue the packet at the head of the transmission queue and extract its length L and drop precedence P.
3) For all values p<=P (from the packet), QLen[p]=QLen [p]−L
4) Deliver the packet to the transmitter for sending
5) Go to step 1) to wait for the transmitter to be ready to accept another packet

The queue between the arrival and transmission processes is a simple FIFO, and the size will be related to Ldrop (at least Ldrop plus enough room for one maximum packet size at every precedence value 0 . . . Pmax, or maybe larger; i.e. minimum queue capacity=Ldrop+(Pmax*<maximum packet size>). For high-speed implementation, the QLen table can be structured as a binary tree of tables, reducing the number of adds and subtracts from an average of Pmax/2 to log 2(Pmax).

A flow chart of particular embodiments of the presently disclosed methods are depicted in FIGS. **10**A, **10**B and **11**. The rectangular elements are herein denoted "processing blocks" and represent computer software instructions or groups of instructions. Alternatively, the processing blocks represent steps performed by functionally equivalent circuits such as a digital signal processor circuit or an application specific integrated circuit (ASIC). The flow diagrams do not depict the syntax of any particular programming language. Rather, the flow diagrams illustrate the functional information one of ordinary skill in the art requires to fabricate circuits or to generate computer software to perform the processing required in accordance with the present invention. It should be noted that many routine program elements, such as initialization of loops and variables and the use of temporary variables are not shown. It will be appreciated by those of ordinary skill in the art that unless otherwise indicated herein, the particular sequence of steps described is illustrative only and can be varied without departing from the spirit of the invention. Thus, unless otherwise stated the steps described below are unordered meaning that, when possible, the steps can be performed in any convenient or desirable order.

Referring now to FIGS. **10**A and **10**B, a method **100** of routing a packet having a drop precedence value is shown. Method **100** begins with processing block **102** which recites receiving a packet having a drop precedence value associated therewith. The packet may be received from a grooming router or from a source which inserts the drop precedence value in the packet. Processing block **104** discloses wherein the receiving a packet having a drop precedence value associated therewith comprises the packet having a drop precedence value in an Internet Protocol (IP) Type of Service (TOS) field of the packet. Two packets in a same flow may have different drop precedence values.

Processing block **106** states determining for the packet whether a sum of queued packet sizes of previously received packets having a higher drop precedence value than the packet is larger than a first threshold value. As recited in processing block **108**, the determining for the packet whether a sum of queued packet sizes of previously received packets having a higher drop precedence value than the packet is larger than a first threshold value utilizes a queue length counter per precedence level.

Processing block **110** discloses dropping the packet when the sum of queued packet sizes of the previously received packets having a higher drop precedence value than the packet is larger than the first threshold value. The packet is forwarded when the sum of queued packet sizes of the previously received packets having a higher drop precedence value than the packet is less than the first threshold value.

Processing block **112** states wherein the determining for the packet whether a sum of queued packet sizes of previously received packets having a higher drop precedence value than the packet is larger than a first threshold value and the dropping the packet when the sum of queued packet sizes of the previously received packets having a higher drop precedence

value than the packet is larger than the first threshold value are used for at least one of the group consisting of providing Service Level Agreement (SLA) guarantees wherein a network provider performs one of the group consisting of dropping traffic exceeding the SLA, and charging additional fees for excess use; performing Transmission Control Protocol (TCP) flow control; propagating cutoff values for a plurality of links of the network and using said cutoff values to determine which packets to drop upstream of congestion points in said network; and performing routing, wherein the threshold value of packets is provided to routers and packets can be re-routed to avoid congestion.

Processing continues with processing block **114** which discloses determining for the packet whether a sum of queued packet sizes of previously received packets having a higher drop precedence value than the packet is larger than a second threshold value.

Processing block **116** states dropping the packet when the sum of queued packet sizes of the previously received packets having a higher drop precedence value than the packet is larger than the second threshold value, wherein the second threshold value is applied for a second queue. As shown in processing block **118**, the second queue is for low-latency traffic. The actions described in processing blocks **114** and **116** provide a PDQoS involving two types of traffic (e.g., low latency traffic and normal traffic).

Referring now to FIG. **11**, a particular embodiment of a method **150** for assigning a drop precedence level to a packet is shown. Method **150** begins with processing block **152** which discloses determining a drop precedence value for a packet. As shown in processing block **154** the determining is performed by either a grooming router, a source host operating system or an application running on the source host. Processing block **156** states at least two packets in a same flow have different drop precedence values. Processing block **158** recites wherein the drop precedence value is capable of being used by a router for determining for the packet whether a sum of queued packet sizes of previously received packets having a higher drop precedence value than the packet is larger than a first threshold value and dropping the packet when the sum of queued packet sizes of the previously received packets having a higher drop precedence value than the packet is larger than the first threshold value.

Processing block **160** recites inserting the drop precedence value into the packet. As shown in processing block **162**, the inserting the drop precedence value into the packet comprises storing the drop precedence value into value in one of the group consisting of the Internet Protocol (IP) Type Of Service (TOS) field of the packet, an Internet Protocol (IP) option and a Transmission Control Protocol (TCP) option.

Processing block **164** discloses transmitting the packet having the drop precedence value inserted therein. The algorithm used to set the precedence value is typically going to be user-configurable, and will often be expressed as set of policy statements to configure the grooming router (or host OS, or application). Policies will often consider the general type of traffic (such as the SLA carve-up in FIG. **9** shows), as well as protocol and application data details (as discussed in the marking examples shown in FIGS. **2** and **3**).

FIG. **12** is a block diagram illustrating an example computer system **200** for implementing PDQoS function **240** and/or other related processes to carry out the different functionality as described herein.

As shown, computer system **200** of the present example includes an interconnect **211** that couples a memory system **212** and a processor **213** an input/output interface **214**, and a communications interface **215**.

As shown, memory system **212** is encoded with PDQoS application **240-1**. PDQoS application **240-1** can be embodied as software code such as data and/or logic instructions (e.g., code stored in the memory or on another computer readable medium such as a disk) that support functionality according to different embodiments described herein.

During operation, processor **213** of computer system **200** accesses memory system **212** via the interconnect **211** in order to launch, run, execute, interpret or otherwise perform the logic instructions of the PDQoS application **240-1**. Execution of PDQoS application **240-1** produces processing functionality in PDQoS process **240-2**. In other words, the PDQoS process **240-2** represents one or more portions of the PDQoS application **240-1** (or the entire application) performing within or upon the processor **213** in the computer system **200**.

It should be noted that, in addition to the PDQoS process **240-2**, embodiments herein include the PDQoS application **240-1** itself (i.e., the un-executed or non-performing logic instructions and/or data). The PDQoS application **240-1** can be stored on a computer readable medium such as a floppy disk, hard disk, or optical medium. The PDQoS application **240-1** can also be stored in a memory type system such as in firmware, read only memory (ROM), or, as in this example, as executable code within the memory system **212** (e.g., within Random Access Memory or RAM).

In addition to these embodiments, it should also be noted that other embodiments herein include the execution of PDQoS application **240-1** in processor **213** as the PDQoS process **240-2**. Those skilled in the art will understand that the computer system **200** can include other processes and/or software and hardware components, such as an operating system that controls allocation and use of hardware resources associated with the computer system **200**.

As described in detail above, PDQoS is a method for achieving prioritized quality of service via a combination of priority assignments to each packet entering a network and a priority-based dropping scheme at congested queues in the network. In PDQoS, packets in a flow are assigned priorities uniformly at random from the priority band for the flow. Prioritization between different flows is achieved by assigning different priority bands to different flows. The expectation is that flows with a higher priority band would achieve higher throughput in the network.

The PDQoS uniform distribution scheme for assigning priorities does not achieve a linear relationship between priority-band-overlap and throughput when the flows use the Transmission Control Protocol (TCP), the most common transport protocol in the Internet.

FIG. **13** shows the resulting priority distribution **300** of simulating four TCP flows with priority bands **302**, **304**, **306** and **308**. Priority band **302** includes priority levels 1-16, priority band **304** includes priority levels 9-24, priority band **306** includes priority levels 17-32, and priority band **308** includes priority levels 25-40. All the priority bands **302**, **304**, **306** and **308** are sharing a 4 Mb/s bottleneck link. The priority bands have a small amount of overlap. As can be seen from FIG. **13**, the assigned priorities are distributed evenly across each priority band.

FIG. **14** shows the resulting throughput distribution **310** of simulating four TCP flows with priority bands **312**, **314**, **316** and **318**. FIG. **14** shows the bandwidth achieved by each flow during the same simulation run as in FIG. **13**, which showed the priorities assigned to the packets. Priority band **312** includes priority levels 1-16, priority band **314** includes priority levels 9-24, priority band **316** includes priority levels 17-32, and priority band **318** includes priority levels 25-40,

all sharing a 4 Mb/s bottleneck link. The priority bands have a small amount of overlap. As can be seen from FIG. **13** and FIG. **14**, priority band **308** and **318** receives a vast majority of the bandwidth, while the remaining priority bands receive a much smaller share.

FIG. **15** shows the resulting priority distribution **320** of simulating four TCP flows with priority bands **322**, **324**, **326** and **328**. Priority band **322** includes priority levels 1-16, priority band **324** includes priority levels 3-18, priority band **326** includes priority levels 5-20, and priority band **328** includes priority levels 7-22 sharing a 4 Mb/s bottleneck link.

FIG. **16** shows the resulting throughput distribution **330** of simulating four TCP flows with priority bands **332**, **334**, **336** and **338**. Priority band **332** includes priority levels 1-16, priority band **334** includes priority levels 3-18, priority band **336** includes priority levels 5-20, and priority band **3338** includes priority levels 7-22, all the priority bands sharing a 4 Mb/s bottleneck link. A similar result in shown in FIGS. **15** and **16** as is seen in FIGS. **13** and **14**, wherein the highest priority bands **318** and **328** have a much higher throughput than the remaining priority bands.

FIGS. **13** and **15** show the number of packets that were randomly assigned to each exact priority within the band. FIGS. **14** and **16** show the throughput achieved over time by each of the four flows in each test. Clearly, the larger priority band overlap in the second test run does not have a large impact on the prioritization of the flows.

Referring now to FIG. **17**, the percent of bottleneck capacity used by each flow by priority band overlap is shown. Eight TCP flows are shown, 4 using one priority band and 4 using another. The x-axis shows the percent of the priority bands that do not overlap. The y-axis shows the throughput divided by the bottleneck capacity for each of the eight flows. This Figure shows the relationship between the amount of priority overlap between the flows and the percentage of the bottleneck capacity that is used by each flow. It can be seen that there is not a linear relationship between priority band overlap and achieved throughput

The non-linear relationship can be explained by considering the behavior of TCP. Whenever a packet is dropped, the TCP sender drops its sending rate by half. When the PDQoS pcutoff value hits a certain priority, all packets at or below that priority will be dropped. Consider two senders, sender 1 using a priority band 1-8, sender 2 using a band 2-9. If the pcutoff value happens to be at priority 2, then sender 2 sees ⅛ of its packets dropped while sender 1 sees ¼ of its packets dropped. Therefore, sender 1 will backoff by half twice as often as sender 2, so during the time it takes for sender 2 to drop its sending rate to (½) times its previous rate, sender 1 will drops its rate to $(½)^2$ of its rate. Therefore, the amount of priority band overlap has an exponential relationship to the amount that the TCP sender will back off. This can be seen in FIG. **17**.

The presently described non-uniform per-packet priority marking for use with adaptive protocols counteracts the problem by using a non-uniform distribution for assigning priorities to TCP flows.

PDQoS has potential to fill the need for a quality of service mechanism that is simple to configure and to understand, inexpensive to use, and beneficial to both ISPs and end users. However, the simplicity of configuring PDQoS depends heavily on being able to define priority bands in such a way to get the desired capacity sharing. As discussed above, the original uniform distribution scheme for assigning priorities in PDQoS makes it difficult (if not impossible) to get fine grained control over how the capacity will be allocated across a series of TCP flows (if all flows should get some of the capacity, but "how much" of the capacity should depend on the relative priorities).

Internet Service Providers (ISPs) are not likely to want all (or even most) of the capacity to go to the highest priority users—they are more likely to want a gradual degradation of throughput as the priority band decreases relative to other bands. The presently described non-uniform per-packet priority marking for use with adaptive protocols gives exactly that: the ability to easily configure PDQoS such that TCP flows can achieve a desired distribution of available capacity.

This invention applies not only to PDQoS with TCP, but also to any packet-level prioritization scheme that assigns priorities to packets randomly or pseudorandomly and to any protocol that adapts to loss.

Given a set of flows and a desired percentage of the bottleneck capacity that should be assigned to each, a user or administrator would assign priority bands to adaptive flows such that the top of the bands are spaced proportionally to the desired capacity sharing. See FIG. **22** for an example of how assigned bands might affect bandwidth sharing.

For flows using a protocol that adapts to congestion, each priority band is specified by only a single number, representing the highest priority that could be assigned to a packet in this flow. Each time a packet needs to be assigned an exact priority, a priority is chosen at random using some non-uniform distribution between some lowest priority (the same for all flows) and the top of the band. This invention covers any non-uniform distribution. The distribution chosen combined with the protocol used will affect the bandwidth sharing.

For example: if priorities are distributed via the bottom half of a normal distribution over a range from 0 to twice the top of the priority band and the adaptive protocol drops its rate in half when packet loss is detected (as in TCP), the relationship between priority band and bottleneck capacity sharing is approximately linear.

To achieve a linear relationship between the priority band and the bottleneck capacity sharing, the amount of reaction per dropped packet should be inversely proportional to the difference between the number of packets assigned to two neighboring priorities. Note that this also fits with non-adaptive protocols using a uniform distribution, as described in the original PDQoS invention.

At each bottleneck queue, an existing prioritization mechanism is used (such as PDQoS) to determine based on the priority whether or not to drop each packet.

If the distribution is inversely proportional to the amount of reaction per dropped packet, the number of packets at the lowest priorities is inverse-exponentially proportionate to the defined top of the priority band. When combined with multiplicative back-off, this means that the average sending rate for each adaptive flow adapts to be linearly proportional to the top of the priority band.

The Figures discussed below show the results of the same simulations as the results discussed in FIGS. **13-16** above, except that the uniform distribution priority dithering has been replaced by normal distribution dithering for TCP, as described in this invention.

FIG. **18** shows the resulting distribution of priorities to packets **400** of simulating four TCP flows with normal distribution priority bands **402**, **404**, **406** and **408** topped at 16, 24, 32, and 40 respectively, all sharing a 4 Mb/s bottleneck link.

FIG. **19** shows the resulting throughput distribution **410** of simulating four TCP flows with normal distribution priority bands **412**, **414**, **416** and **418** topped at 16, 24, 32, and 40 respectively, all sharing a 4 Mb/s bottleneck link. As can be

seen from FIG. **18** and FIG. **19**, priority band **408** and **418** still receives a majority of the bandwidth, while the remaining three priority bands also receive a fair share of the remaining bandwidth.

FIG. **20** shows the resulting distribution of priorities to packets **420** of simulating four TCP flows with normal distribution priority bands **422**, **424**, **426** and **428** topped at 16, 18, 20, and 22 respectively, all sharing a 4 Mb/s bottleneck link.

FIG. **21** shows the resulting throughput distribution **430** of simulating four TCP flows with normal distribution priority bands **432**, **434**, **436** and **438** topped at 16, 18, 20, and 22 respectively, all sharing a 4 Mb/s bottleneck link. As can be seen from FIG. **20** and FIG. **21**, priority band **408** and **418** still receives a majority of the bandwidth, while the remaining three priority bands also receive a fair share of the remaining bandwidth.

FIGS. **18** and **20** show the number of packets that were randomly assigned to each exact priority within the band. FIGS. **19** and **21** show the throughput achieved over time by each of the four flows in each test. Unlike the results without this invention, the larger priority band overlap in the second test run has a larger impact on the prioritization of the flows.

Referring now to FIG. **22**, a percent of bottleneck capacity used by each flow by priority band overlap is shown. Eight TCP flows are shown, four flows using one priority band and four using another. The x-axis shows the percent of priority bands that do not overlap. The y-axis shows the throughput divided by the bottleneck capacity for each of the eight flows. Comparing this figure to FIG. **17**, which shows the same test with a uniform distribution, shows the improvement in prioritization gained by using the invention, since the relationship between band overlap and throughput is now almost exactly linear, as desired.

Referring now to FIG. **23**, a particular embodiment of a method **500** for non-uniform per-packet priority marking for use with adaptive protocols is shown. Method **500** begins with processing block **502** which discloses receiving a packet at a first network device, the packet assigned to a priority band. Processing block **504** states determining a priority for the packet between a lowest priority of the priority band and a highest priority of the priority band, the priority for the packet selected based on a target distribution of priorities within the priority band, the target distribution comprising a distribution selected to achieve a desired capacity relationship among groups of packets assigned to different priority bands. As shown in processing block **506**, in one embodiment the target distribution is selected to approximate a linear relationship between a priority band and capacity share. As shown in processing block **508**, in certain embodiments, the target distribution comprises a non-uniform distribution. As shown in processing block **510**, in certain embodiments, the target distribution is based on a reaction rate of a protocol used for transporting the packet. As further shown in processing block **512**, in some embodiments the protocol may comprise a protocol that adapts to congestion. As described in processing block **514**, in a particular embodiment, a number of packets at a lowest priority of the priority band are inverse-exponentially proportionate to a number of packets at a highest priority of the priority band.

Processing block **516** discloses assigning the selected priority to the packet. Processing continues at processing block **518** which recites using a prioritization mechanism to determine, based on the assigned priority of the packet, whether to drop the packet. As shown in processing block **518**, in some embodiments, the prioritization mechanism comprises a Precedence Drop Quality of Service (PDQoS).

FIG. **24** is a block diagram illustrating an example computer system **600** for implementing per-packet priority marking function **240** and/or other related processes to carry out the different functionality as described herein.

As shown, computer system **200** of the present example includes an interconnect **211** that couples a memory system **212** and a processor **213** an input/output interface **214**, and a communications interface **215**.

As shown, memory system **212** is encoded with Per-packet priority marking application **240-1**. Per-packet priority marking application **240-1** can be embodied as software code such as data and/or logic instructions (e.g., code stored in the memory or on another computer readable medium such as a disk) that support functionality according to different embodiments described herein.

During operation, processor **213** of computer system **200** accesses memory system **212** via the interconnect **211** in order to launch, run, execute, interpret or otherwise perform the logic instructions of the Per-packet priority marking application **240-1**. Execution of Per-packet priority marking application **240-1** produces processing functionality in Per-packet priority marking process **240-2**. In other words, the Per-packet priority marking process **240-2** represents one or more portions of the Per-packet priority marking application **240-1** (or the entire application) performing within or upon the processor **213** in the computer system **200**.

It should be noted that, in addition to the Per-packet priority marking process **240-2**, embodiments herein include the Per-packet priority marking application **240-1** itself (i.e., the unexecuted or non-performing logic instructions and/or data). The Per-packet priority marking application **240-1** can be stored on a computer readable medium such as a floppy disk, hard disk, or optical medium. The Per-packet priority marking application **240-1** can also be stored in a memory type system such as in firmware, read only memory (ROM), or, as in this example, as executable code within the memory system **212** (e.g., within Random Access Memory or RAM).

In addition to these embodiments, it should also be noted that other embodiments herein include the execution of Per-packet priority marking application **240-1** in processor **213** as the Per-packet priority marking process **240-2**. Those skilled in the art will understand that the computer system **200** can include other processes and/or software and hardware components, such as an operating system that controls allocation and use of hardware resources associated with the computer system **200**.

The device(s) or computer systems that integrate with the processor(s) may include, for example, a personal computer(s), workstation(s) (e.g., Sun, HP), personal digital assistant(s) (PDA(s)), handheld device(s) such as cellular telephone(s), laptop(s), handheld computer(s), packet-switching device such as an IP or MPLS router or an Ethernet switch, or another device(s) capable of being integrated with a processor(s) that may operate as provided herein. Accordingly, the devices provided herein are not exhaustive and are provided for illustration and not limitation.

References to "a microprocessor" and "a processor", or "the microprocessor" and "the processor," may be understood to include one or more microprocessors that may communicate in a stand-alone and/or a distributed environment(s), and may thus be configured to communicate via wired or wireless communications with other processors, where such one or more processor may be configured to operate on one or more processor-controlled devices that may be similar or different devices. Use of such "microprocessor" or "processor" terminology may thus also be understood to include a central processing unit, an arithmetic logic unit, an application-spe-

cific integrated circuit (IC), and/or a task engine, with such examples provided for illustration and not limitation.

Furthermore, references to memory, unless otherwise specified, may include one or more processor-readable and accessible memory elements and/or components that may be internal to the processor-controlled device, external to the processor-controlled device, and/or may be accessed via a wired or wireless network using a variety of communications protocols, and unless otherwise specified, may be arranged to include a combination of external and internal memory devices, where such memory may be contiguous and/or partitioned based on the application. Accordingly, references to a database may be understood to include one or more memory associations, where such references may include commercially available database products (e.g., SQL, Informix, Oracle) and also proprietary databases, and may also include other structures for associating memory such as links, queues, graphs, trees, with such structures provided for illustration and not limitation.

References to a network, unless provided otherwise, may include one or more intranets and/or the Internet, as well as a virtual network. References herein to microprocessor instructions or microprocessor-executable instructions, in accordance with the above, may be understood to include programmable hardware.

Unless otherwise stated, use of the word "substantially" may be construed to include a precise relationship, condition, arrangement, orientation, and/or other characteristic, and deviations thereof as understood by one of ordinary skill in the art, to the extent that such deviations do not materially affect the disclosed methods and systems.

Throughout the entirety of the present disclosure, use of the articles "a" or "an" to modify a noun may be understood to be used for convenience and to include one, or more than one of the modified noun, unless otherwise specifically stated.

Elements, components, modules, and/or parts thereof that are described and/or otherwise portrayed through the figures to communicate with, be associated with, and/or be based on, something else, may be understood to so communicate, be associated with, and or be based on in a direct and/or indirect manner, unless otherwise stipulated herein.

Although the methods and systems have been described relative to a specific embodiment thereof, they are not so limited. Obviously many modifications and variations may become apparent in light of the above teachings. Many additional changes in the details, materials, and arrangement of parts, herein described and illustrated, may be made by those skilled in the art.

Having described preferred embodiments of the invention it will now become apparent to those of ordinary skill in the art that other embodiments incorporating these concepts may be used. Additionally, the software included as part of the invention may be embodied in a computer program product that includes a computer useable medium. For example, such a computer usable medium can include a readable memory device, such as a hard drive device, a CD-ROM, a DVD-ROM, or a computer diskette, having computer readable program code segments stored thereon. The computer readable medium can also include a communications link, either optical, wired, or wireless, having program code segments carried thereon as digital or analog signals. Accordingly, it is submitted that that the invention should not be limited to the described embodiments but rather should be limited only by the spirit and scope of the appended claims.

What is claimed is:

1. A method comprising:

receiving, at a first network device, a packet in a first flow of a plurality of flows, said packet being assigned to a priority band, said priority band including a lowest priority and a highest priority;

selecting a target distribution of priorities within said first flow to achieve a desired capacity relationship across said plurality of flows;

selecting, for said packet in said first flow, a priority between said lowest priority and said highest priority of said priority band based on said target distribution of priorities within said first flow; and

assigning said selected priority to said packet.

2. The method of claim **1** wherein said target distribution is selected to approximate a linear relationship between a priority band and capacity share.

3. The method of claim **1** wherein said target distribution comprises a non-uniform distribution.

4. The method of claim **1** wherein said target distribution is based on a reaction rate of a protocol used for transporting said packet.

5. The method of claim **4** wherein said protocol comprises a protocol that adapts to congestion.

6. The method of claim **1** wherein a number of packets at said lowest priority of said priority band is inverse-exponentially proportionate to a number of packets at said highest priority of said priority band.

7. The method of claim **1** further comprising using a prioritization mechanism to determine, based on the assigned priority of said packet, whether to drop said packet.

8. The method of claim **7** wherein said prioritization mechanism comprises a Precedence Drop Quality of Service (PDQoS).

9. A non-transitory computer readable storage medium having computer readable code thereon for non-uniform per-packet priority marking for use with adaptive protocols, the medium including instructions in which a computer system performs operations comprising:

receiving, at a first network device, a packet in a first flow of a plurality of flows, said packet being assigned to a priority band, said priority band including a lowest priority and a highest priority;

selecting a target distribution of priorities within said first flow to achieve a desired capacity relationship across said plurality of flows;

selecting, for said packet in said first flow, a priority between said lowest priority and said highest priority of said priority band based on said target distribution of priorities within said first flow; and

assigning said selected priority to said packet.

10. The non-transitory computer readable storage medium of claim **9** wherein said target distribution is selected to approximate a linear relationship between a priority band and capacity share.

11. The non-transitory computer readable storage medium of claim **9** wherein said target distribution comprises a non-uniform distribution.

12. The non-transitory computer readable storage medium of claim **9** wherein said target distribution is based on a reaction rate of a protocol used for transporting said packet.

13. The non-transitory computer readable storage medium of claim **12** wherein said protocol comprises a protocol that adapts to congestion.

14. The non-transitory computer readable storage medium of claim **9** wherein a number of packets at said lowest priority

of said priority band is inverse-exponentially proportionate to a number of packets at said highest priority of said priority band.

15. The non-transitory computer readable storage medium of claim **9** further comprising using a prioritization mechanism to determine, based on the assigned priority of said packet, whether to drop said packet.

16. The non-transitory computer readable storage medium of claim **15** wherein said prioritization mechanism comprises a Precedence Drop Quality of Service (PDQoS).

17. A computer system comprising:

a memory;

a processor;

a communications interface; and

an interconnection mechanism coupling the memory, the processor and the communications interface,

wherein the memory is encoded with an application providing non-uniform per-packet priority marking for use with adaptive protocols that, when performed on the processor, provides a process for processing information, the process causing the computer system to perform the operations of:

receiving, at a first network device, a packet in a first flow of a plurality of flows, said packet being assigned to a priority band, said priority band including a lowest priority and a highest priority;

selecting a target distribution of priorities within said first flow to achieve a desired capacity relationship across said plurality of flows;

selecting, for said packet in said first flow, a priority between said lowest priority and said highest priority of said priority band based on said target distribution of priorities within said first flow; and

assigning said selected priority to said packet.

18. The computer system of claim **17** wherein said target distribution is selected to approximate a linear relationship between a priority band and capacity share.

19. The computer system of claim **17** wherein a number of packets at said lowest priority of said priority band is inverse-exponentially proportionate to a number of packets at said highest priority of said priority band.

20. The computer system of claim **17** further comprising using a prioritization mechanism to determine, based on the assigned priority of said packet, whether to drop said packet.

* * * * *